technische universität
dortmund

Masterarbeit

# Fehlererkennung durch Unsicherheitsschätzung mit Tiefen Neuronalen Netzen in Industrie 4.0

Lucas Weiße
26. Mai 2021

Gutachter:
Prof. Dr. Katharina Morik
Sebastian Buschjäger

Technische Universität Dortmund
Fakultät für Informatik
Lehrstuhl für Künstliche Intelligenz (LS-8)
https://www-ai.cs.tu-dortmund.de/index.html

# Inhaltsverzeichnis

# Symbolverzeichnis

**Abkürzungen / Akronyme**

| | |
|---|---|
| Abb. | Abbildung |
| BS | Brier Score |
| bspw. | beispielsweise |
| CNN | Faltendes Neuronales Netzwerk/ convolutional neural network |
| d.h. | das heißt |
| ECE | Expected Calibration Error |
| FC | vollständig verbundene Schicht/ fully connected layer |
| H | Entropie |
| I | Gegenseitige Information/ mutual information |
| MC-Dropout | Monte Carlo Dropout |
| NLL | Negativer Log-Likelihood |
| TACE | Threshold Adaptive Calibration Error |
| u.a. | unter anderem |
| z.B. | zum Beispiel |

**Griechische Symbole**

| | |
|---|---|
| $\alpha$ | Skalenparameter |
| $\beta$ | Bias oder Shift-Parameter |
| $\chi$ | Identitätsfunktion |
| $\delta$ | Lernratenverlust |

| | |
|---|---|
| $\gamma$ | Grenzwert |
| $\lambda$ | Gewichtsverlust |
| $\omega$ | Gewichtsvektor |
| $\sigma$ | nichtlineare Aktivierungsfunktion |
| $\theta$ | Menge an Zufallsvariablen (zum Beispiel Modellparameter) |

**Messrollen Signal Simulator (Parameter im Simulator)**

| | |
|---|---|
| $\varphi_{Entry}$ | Eingangswinkel des Bandes auf der Planheitsmessrolle (ROLL.phi_en) |
| $\varphi_{Exit}$ | Ausgangswinkel des Bandes auf der Planheitsmessrolle (ROLL.phi_ex) |
| $B_{Misalign}$ | Axiale Verschiebung des Bandes (Band.Misalignment) |
| $DG_{Amplitude}$ | Dirty Gap Amplitudenstärke (errorData.DirtyGap.Amplitude) |
| $DS_{Teil}$ | Datensatzgröße bei der Teilüberdeckung |
| $DS_{Voll}$ | Datensatzgröße bei der Vollüberdeckung |
| $N_{Power}$ | Rauschstärke (errorData.Noise.Power) |
| $PK_{Teil}$ | Anzahl an Parameterkombinationen bei der Teilüberdeckung |
| $PK_{Voll}$ | Anzahl an Parameterkombinationen bei der Vollüberdeckung |
| $SE_{Frequenz}$ | Sensorfehler Grenzfrequenz (errorData.Sensor.CutOffFrequency) |
| $SE_{Sensor}$ | Vektor der defekten Sensoren (errorData.Sensor.SensorNumber) |
| $SIKO_{Degree}$ | SIKO Fehlerposition (errorData.Siko.ErrorDegree) |
| $w_{Band}$ | Bandbreite (BAND.Width) |

**Römische Symbole**

| | |
|---|---|
| $\hat{y}_n$ | Die vorhergesagte Klasse/Modellausgabe bei Eingabe $x_n$ |
| $\mathcal{D}$ | Datensatz |
| $\mathcal{T}$ | Temperaturskalar |
| $x_n$ | Ein Modell Eingabewert/ Zeitreihe |
| $y$ | (Modell-)Ausgabe |
| $y_n$ | Das Klassenlabel der Eingabe $x_n$ |

| | |
|---|---|
| $z_n$ | Die Modell Logits bei Eingabe $x_n$ |

**Operatoren**

| | |
|---|---|
| $\langle \cdot, \cdot \rangle$ | Skalarprodukt |
| $\odot$ | elementweises Produkt |

**Zahlenbereiche**

| | |
|---|---|
| $\mathbb{R}$ | Reelle Zahlen |

# Kapitel 1

# Einleitung

## 1.1 Motivation und Hintergrund

In den letzten Jahren gab es viele Fortschritte im Bereich des Maschinellen Lernens. Schon länger werden automatisierte, informationsverarbeitende Systeme erfolgreich in realen Anwendungen eingesetzt. Oft handelt es sich dabei um Anwendungsbereiche mit wenig Risiko, bei denen Fehlentscheidungen eine akzeptable Tragreichweite besitzen. Ein jüngerer Trend ist der Einsatz automatisierter Systeme in risikobehafteten Bereichen, die sogar lebensgefährlich für Menschen seien können. Beispielsweise in Anwendungen mit potenziell hohen finanziellen Schäden, bei selbstfahrenden Autos und medizinischen Empfehlungssystemen. In risikoreichen Anwendungsbereichen müssen Systeme in der Lage sein, die eigene Unsicherheit zu identifizieren, damit menschliche Hilfe angefordert oder automatisierte Entscheidungen eingegrenzt werden können.

Neuronale Netzwerke sind eine Methode des überwachten Lernens, die sowohl in der Klassifikation, als auch in der Regression eingesetzt werden. Strukturell bestehen Neuronale Netze aus gewichteten Summen und nichtlinearen Transformationen, die versuchen, zugrunde liegende Beziehungen in einem Datensatz zu erkennen. Inspiriert sind sie von der Funktionsweise des menschlichen Gehirns. Neuronale Netzwerke sollten möglichst gut kalibriert sein, d.h. die probabilistischen Vorhersagen entsprechen den tatsächlichen Wahrscheinlichkeiten. Gut kalibrierte Vorhersagen sind nicht nur oft eine Voraussetzung für weiterverarbeitende Methoden, sondern erleichtern auch die Interpretation der Ergebnisse. Neuronale Netze erreichen hohe Genauigkeiten, sind jedoch schlecht darin ihre Vorhersageunsicherheit zu berücksichtigen. Übermütige falsche Vorhersagen können in risikoreichen Anwendungsbereichen zu potenziellen Schäden führen. Es ist deshalb wünschenswert, dass ein Neuronales Netzwerk neben genauen Vorhersagen, auch seine eigene Unsicherheit bestimmen kann. Generell handelt es sich bei den Parametern und Vorhersagen um Punktschätzer[12]. Die Punktschätzung eines (einzigen) Neuronalen Netzes reicht nicht aus um die Unsicherheit zu bestimmen. Einen Ansatz bieten Bayessche Neuronale Netzwerke,

hier werden die Parameter und Ausgaben durch Wahrscheinlichkeitsverteilungen ersetzt. Leider ist die genaue Bayessche Inferenz für Neuronale Netzwerke rechnerisch nicht umsetzbar, weshalb verschiedene probabilistische, nicht Bayessche Annäherungen entwickelt wurden. Beispielsweise kann man die Vorhersagen mehrerer Neuronaler Netze in einem Ensemble zusammenfassen und als Bereichsschätzer interpretieren[19]. Ein weiterer Ansatz ist der Monte Carlo Dropout[11], hier wird für jede Vorhersage das Modell zufällig verändert. Für eine Eingabe werden dann mehrere Vorhersagen berechnet und anschließend zusammengefasst. Dieser Vorgang ähnelt dem Ziehen aus einer Verteilung.

In dieser Arbeit soll das Signal einer Planheitsmessrolle auf unterschiedliche Fehlerszenarien untersucht und anschließend klassifiziert werden. Eine Planheitsmessrolle misst die Ebenheit eines gewalzten Bandes. Dabei werden mehrere Sensoren Zonenweise eingeteilt, messen den Druck des Bandes auf der Rolle und fassen die Informationen in Kanälen zusammen. Aufgrund der schnellen Rotationsgeschwindigkeit der Planheitsmessrolle und der hohen Anzahl an Sensoren werden sehr schnell große Datenmengen erstellt. Das langfristige, kontinuierliche Speichern dieser Mengen ist nicht umsetzbar, weshalb ein Algorithmus die Daten während des Prozesses analysieren soll. Der Analysevorgang soll neben dem Normalzustand, vier Fehlerzustände erkennen und entsprechend klassifizieren. Die Ausgabesignale der Planheitsmessrolle können als eine Zeitreihe interpretiert werden. Die Klassifikation von Zeitreihen ist aufgrund ihrer Eigenschaften, wie z.B. großen Datenmengen oder hohen Dimensionalitäten, ein anspruchsvolles Problem. Traditionell mussten Merkmale aus kleinen Ausschnitten einer Zeitreihe extrahiert werden. Dieser Vorgang ist nicht einfach und erfordert das Wissen eines Experten. Die Merkmale wurden anschließend in Entscheidungsbäume oder Ensembles verwendet. Ein neuerer Ansatz ist die Nutzung von Neuronalen Netzen. Rekurrente und eindimensionale Faltende Neuronale Netzwerke erreichen bei der Zeitreihenklassifikation beeindrucke Ergebnisse. Ein Vorteil bei dieser Herangehensweise ist, dass die Merkmale aus den Daten extrahiert werden und kein Expertenwissen nötig ist.

In der Lernphase benötigten einige Algorithmen des Maschinellen Lernens Beispieldaten. Diese Daten stellen neben den unterschiedlichen Eingaben, auch die gewünschten Ausgaben bereit. In der realen Welt ist das Sammeln solcher Daten meistens schwer. Sie müssen nicht nur qualitativ gut sein, sondern auch sehr umfangreich. Normalerweise funktionieren die Algorithmen besser, wenn beim Training viele Daten zur Verfügung standen. In dieser Arbeit kann ein Simulator zur Generierung der Beispieldaten verwendet werden. Der Vorteil eines Simulators ist, dass sich schneller und einfacher umfangreiche Datensätze erstellen lassen. Die Daten können anschließend beim Training des Modells genutzt werden. Ziel ist dabei, dass sich die gelernten Eigenschaften auf das reale Problem anwenden lassen.

## 1.2 Ziel der Arbeit

In dieser Arbeit soll das Signal einer Planheitsmessrolle durch ein Faltendes Neuronales Netzwerk auf unterschiedliche Fehlerszenarien untersucht und entsprechend klassifiziert werden. Die nötigen Trainingsdaten werden mithilfe eines Simulators erstellt. Für den gegebenen Anwendungszweck soll der Simulator zusätzlich erweitert und angepasst werden. Bei der Erstellung des Datensatzes sollen möglichst viele Fälle beachtet werden, damit realitätsnahe Daten beim Training und bei den Experimenten zur Verfügung stehen. Neben der Klassifikation soll untersucht werden, ob das Modell kalibrierte Vorhersagen erstellt und seine eigene Unsicherheit berücksichtigen kann.

## 1.3 Verwandte Arbeiten

Die Klassifikation von Zeitreihen durch ein CNN wurde bereits in verschiedenen Arbeiten untersucht[30][9]. Das in dieser Arbeit entwickelte Modell ist im Kontext von Industrie 4.0 und dementsprechend auf den Anwendungsfall angepasst. Es variiert in seinem Aufbau und seiner Komplexität von den vorgestellten Beispielmodellen[30][9]. Die Berücksichtigung der Modellunsicherheit wurde von Brando et al.[4] im Kontext einer Zeitreihenanalyse untersucht. Über ein Programm sollen verschiedene Funktionen für digitales Geld Management bereit gestellt werden. Die Anwendung soll u.a. die erwarteten Ausgaben und Einnahmen der Kunden schätzen. Die Arbeit unterscheidet sich aber insofern, dass keine Klassifikation durchgeführt und kein CNN verwendet wurde. Gundersen et al.[13] versuchen die Erkennbarkeit der Freisetzung von Meeresgasen zu erhöhen. Dabei nutzen sie ein CNN in Kombination mit MC-Dropout. Das Neuronale Netzwerk soll ebenfalls die eigene Unsicherheit berücksichtigen, behandelt aber ein binäres Problem. Das Modell muss nur zwischen einer Leck- und Nichtleck-Situation unterscheiden. In dieser Arbeit muss das Netzwerk zwischen 5 Fällen unterscheiden. Zusätzlich wird in der Verwendung von Ensembles, ein weiterer Ansatz zur Berücksichtigung der Vorhersageunsicherheit genutzt.

## 1.4 Aufbau der Arbeit

Zunächst werden im Kapitel 2 allgemeine Grundlagen gegeben. Es wird ein Überblick über das Maschinelle Lernen und Bayessche Verfahren gegeben. Zusätzlich wird man in den betrachteten Anwendungsprozess eingeführt. Im Kapitel 3 werden verschiedene technische Grundlagen vorgestellt. Dazu gehören u.a. die Definition der Problemstellung und die Vorstellung von Komponenten des verwendeten Modells. Mehrere Unterkapitel enthalten Informationen über die Vorhersageunsicherheit in Neuronalen Netzen. Dabei werden u.a. Regeln zur Bewertung und Verbesserung von probabilistischen Vorhersagen, sowie Funktionen zur Quantifizierung der Vorhersageunsicherheit vorgestellt. Mit Deep Ensembles und

Monte Carlo Dropout werden zwei Methoden zur Berücksichtigung der Unsicherheit beschrieben. Als letztes wird die verwendete Modellarchitektur vorgestellt. Im Kapitel 4 wird der Datenerhebungsprozess erläutert. Zunächst werden die am Simulator vorgenommenen Erweiterungen beschrieben. Anschließend wird der Leser über verschiedene Eigenschaften der Planheitsmessrolle und des erzeugten Signals aufgeklärt. Diese Informationen sind hilfreich, um die gewählten Parameterkonfigurationen und anderen Einstellungen bzgl. des Datensatzes nachvollziehen zu können. Anschließend wird die Grund- und Fehlerkonfiguration der einzelnen Klassen beschrieben. Zusätzlich wird darauf eingegangen, wie und warum die Datensatzgrößen der Klassen zustande kommen. Im Kapitel 5 werden verschiedene Metriken vorgestellt. Die Experimente und Ergebnisse zur Überprüfung der Vorhersagequalität, sowie zur Schätzung der Modellunsicherheit werden im Kapitel 6 präsentiert. In dem Kapitel 7 ist das Fazit der Arbeit.

# Kapitel 2

# Allgemeine Grundlagen

## 2.1 Maschinelles Lernen

Maschinelles Lernen ist ein Oberbegriff für die Generierung von Wissen aus Erfahrung und Daten. Der Bereich ist eine Untermenge der künstlichen Intelligenz. Algorithmen erstellen ein statistisches Modell, dass darauf trainiert ist, Muster und Korrelationen in Daten zu finden. Die im Training verwendeten Daten sind Teil eines Trainingsdatensatzes. Ziel ist, dass die in der Lernphase gefundenen Strukturen und Gesetzmäßigkeiten der Trainingsdaten, sich auch auf unbekannte Daten[1] sinnvoll anwenden lassen. Im Maschinellen Lernen lassen sich die Ansätze und Lernmodelle normalerweise in die folgenden Kategorien unterteilen: *überwachtes*, *unüberwachtes*, *teilüberwachtes* und *bestärkendes Lernen*. Im Kontext dieser Arbeit ist das überwachte Lernen wichtig, es wird im folgenden Unterkapitel ausführlicher erklärt.

**Überwachtes Lernen:** Dem Algorithmus stehen Paare von Ein- und Ausgaben zur Verfügung. Die Ausgabe ist die Zielvariable, also der gewünschte Wert. In der Trainingsphase lernt der Algorithmus Korrelationen, Unterschiede und andere Strukturen zwischen den Ein- und Ausgaben herzustellen.

**Unüberwachtes Lernen:** Dem Algorithmus stehen keine Zielvariablen zur Verfügung. Er muss selbständig Strukturen und Korrelationen in den Daten finden und identifizieren. Anschließend reagiert der Algorithmus bei neuen Daten auf das Vorhandensein oder Fehlen solcher Eigenschaften.

**Teilüberwachtes Lernen:** Diese Kategorie fällt zwischen das überwachte und unüberwachte Lernen. Nur für einen (kleinen) Teil der Eingaben sind die dazugehörigen Ausgaben bekannt. Sie dienen beim Training als Starthilfe, um Strukturen zu finden, die auf die restlichen Eingaben angewendet werden können.

[1] Die Trainingsdaten und die unbekannten Daten stammen normalerweise aus der gleichen Verteilung.

**Bestärktes Lernen:** Es gibt eine Reihe von erlaubten Aktionen, Regeln und möglichen Endzuständen. Der Algorithmus muss ein bestimmtes Ziel verfolgen und interagiert dabei mit einer (dynamischen) Umgebung. Die Aktionen werden im Kontext seiner aktuellen Situation belohnt oder bestraft. Der Algorithmus versucht die Belohnungen zu maximieren.

### 2.1.1 Überwachtes Lernen

Beim überwachten Lernen stehen dem Algorithmus neben den Eingaben, auch die gewünschten Ausgaben (*Zielvariable*) zur Verfügung. Die Eingabe-Ausgabe-Paare nennt man Beispieldaten. Sie werden in einem Beispieldatensatz gespeichert. Der Beispieldatensatz wird in der Regel in 2-3 separate Datensätze unterteilt: Trainings-, (Validierungs-) und Testdatensatz. Während der Lernphase verwendet der Algorithmus die Trainingsdaten als Grundlage und verbessert das statistische Modell durch die Optimierung einer *Verlustfunktion*. Die Verlustfunktion ordnet jeder Entscheidung des Modells einen Schaden zu. Dabei werden Fehlentscheidungen härter bestraft, als gute Entscheidungen, die der Zielvariable ähnlich sind. Je größer der Schaden ist, desto mehr werden die Parameter des Modells angepasst. Bei der Optimierung muss darauf geachtet werden, dass die gelernte Funktion nicht zu sehr auf die Trainingsbeispiele angepasst ist. In diesem Fall spricht man von einer *Überanpassung*. Zur Kontrolle verwendet man Testdaten. Sie stammen aus der gleichen Verteilung wie die Trainingsdaten und werden beim Lernen nicht genutzt, d.h. der Algorithmus kennt die Daten nicht. Die Testdaten können während/nach der Trainingsphase die Vorhersagequalität überprüfen. *Hyperparameter* beeinflussen den Trainingsverlauf, auch auf ihnen ist eine Überanpassung möglich. Deshalb können Validierungsdaten für ihre Optimierung verwendet werden. Die Herausforderung beim überwachtem Lernen ist die *Generalisierung*, d.h. das Modell macht auch auf neuen, unbekannten Daten gute Vorhersagen.

## 2.2 Bayessche Grundlage

Die formale Sprache der Unsicherheit ist die Bayessche Wahrscheinlichkeitstheorie. Hier werden Wahrscheinlichkeiten durch die Quantifizierung des persönlichen Glaubens bestimmt. Mittels Bayesscher probabilistischer Modellierung, auch *Bayessche Inferenze* genannt, kann man diesen Glauben ausdrücken.
Gegeben sind die Daten $\mathcal{D}$ und ein Modell $m$, wobei $\theta$ die Parameter des Modells sind. In der Bayesschen Inferenz sucht man generell zwei Wahrscheinlichkeitsverteilungen, die A-Posterior-Parameterverteilung und die Vorhersageverteilung für unbekannte Daten. Über

die A-Posteriori-Verteilung werden die Parameter gesucht, welche die Daten gut beschreiben. Die Verteilung folgt aus dem Satz von Bayes:

$$\overbrace{p(\theta|\mathcal{D},m)}^{\text{Posterior}} = \frac{\overbrace{p(\mathcal{D}|\theta,m)}^{\text{Likelihood}}\overbrace{p(\theta|m)}^{\text{Prior}}}{\underbrace{p(\mathcal{D}|m)}_{\text{Marginal Likelihood}}}.$$

Die A-Priori-Verteilung beschreibt die initialen Annahmen, d.h. den Glauben bevor die Observationen $\mathcal{D}$ gemacht worden sind. Der Likelihood ist die Wahrscheinlichkeit die neuen Daten zu observieren, unter der Bedingung, dass die Hypothese wahr ist. Der Marginal Likelihood, auch Modellbeweis genannt, ist die Wahrscheinlichkeit die Daten unter allen möglichen Hypothesen zu observieren:

$$p(\mathcal{D}|m) = \int p(\mathcal{D}|\theta,m)p(\theta|m)d\theta.$$

Leider ist der A-Posteriori oft nicht lösbar, weshalb man auf Schätzungen durch *variational Inferenz* zurückgreift.

## 2.3 Walzwerkanlage und Planheitsmessrolle

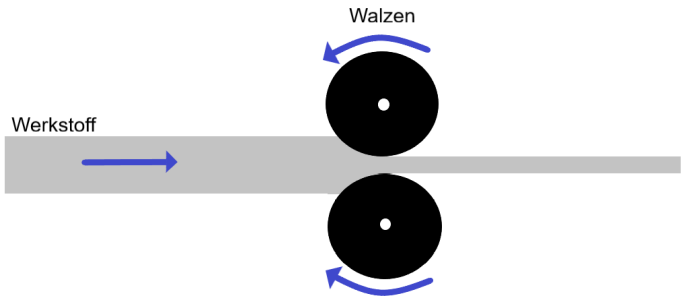
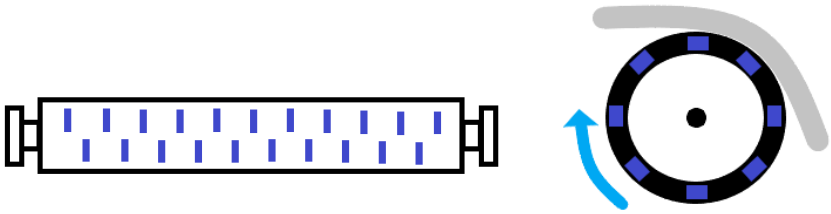



**Abbildung 2.1:** Schematische Visualisierung des Walzverfahren.

Walzwerkanlagen werden in der Metallbearbeitung eingesetzt. Sie verwenden das Fertigungsverfahren Walzen, aus der Gruppe des Druckumformens. Der metallische Werkstoff wird durch ein oder mehrere rotierende Walzpaare geleitet. Entweder soll die Dicke verringert, die Dicke gleichmäßig gemacht oder dem Material eine andere mechanische Eigenschaft verliehen werden. Eine schematische Visualisierung des Walzverfahren ist in Abbildung 2.1 gegeben. Walzen werden u.a. zur Herstellung von Platten, Blechen und Folien genutzt. Eine Planheitsmessrolle misst die Ebenheit eines gewalzten Bandes. Die Ebenheit ist ein wichtiges geometrisches Attribut der Formtoleranz. Pin et al.[24] beschreiben die Ebenheit als das Ausmaß der geometrischen Abweichung von einer Bezugsebene. Die Ursache der

Abweichung ist das Ergebnis einer Werkstückrelaxation nach dem Kalt- oder Warmwalzen. Dabei können innere Spannungsmuster durch die ungleichmäßige Querdruckwirkung der Walzen und die geometrischen und physikalisch Eigenschaften des Eintrittsmaterials verändert werden[24]. Die Ebenheit muss innerhalb bestimmter Toleranzgrenzen bleiben, damit die Bearbeitbarkeit der Bänder bei folgenden Verarbeitungsprozessen gewährleistet werden kann. Bei der Kontrolle wird der Werkstoff über eine rotierende Planheitsmessrolle geleitet. In Abbildung 2.2 ist eine Planheitsmessrolle schematisch dargestellt. Zusätzlich werden im Anhang (Abb. 8.1 und 8.2) detailliertere Schemata bereit gestellt. Auf der Messrolle sind flächendeckend Piezoelektrische Sensoren angebracht. Sie messen an verschiedenen Stellen den ausgeübten Druck des Bandes und geben einen elektrischen Spannungswert weiter.



**(a)** Seitenansicht **(b)** Queransicht mit Band

**Abbildung 2.2:** Schematische Darstellung einer Planheitsmessrolle (Die Anzahl und Anordnung der Sensoren [blaue Rechtecke] sind zufällig gewählt).

# Kapitel 3

# Technische Grundlagen

## 3.1 Klassifikation einer Zeitreihe und Problemstellung

Eine Zeitreihe ist eine zeitliche Abfolge von Daten. Im Kontext der Arbeit handelt es sich um eine Reihe von Messungen. Abhängig davon ob man eine einzige oder mehrere Zeitreihen gleichzeitig betrachtet, unterscheidet man zwischen einer *univariaten* und *multivariaten Zeitreihe*. Eine univariate Zeitreihe kann als ein Vektor $x = (a_1, a_2, \ldots, a_T)$ definiert werden. Die Länge des Vektors ist äquivalent zur Anzahl der Zeit- oder Messpunkte $T$. Eine multivariate Zeitreihe ist eine Matrix $x \in \mathbb{R}^{Z \times T}$, sie enthält $Z$ univariate Zeitreihen. Die Beispieldaten sind eine Menge $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, wobei ein Paar $(x_n, y_n)$ aus einer uni- oder multivariaten Zeitreihe $x_n$ und dem Klassenlabel $y_n$ besteht. Das Klassenlabel wird normalerweise durch eine *One-Hot-Kodierung* dargestellt.

In dieser Arbeit wird ein Mehrklassenproblem betrachtet, ein $T-$dimensionaler Feature Vektor $x \in \mathbb{R}^T$ soll einem Klassenlabel $y \in \{1, \ldots, K\}$ zugeordnet werden. Anhand eines Trainingsdatensatzes mit i.i.d. Daten $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$, soll ein Neuronales Netzwerk die probabilistische Vorhersageverteilung $p_\theta(y|x)$ lernen. $\theta$ bezeichnet die Parameter des Modells. Falls eine Modellkombination (z.B. Ensemble) betrachtet wird, spezifiziert $M$ die Anzahl der Modelle und $\{\theta_m\}_{m=1}^M$ die Parameter des jeweiligen Modells. In einem Netzwerk wird die Vorhersageverteilung durch die Softmax-Funktion $\sigma(z)$ definiert:

$$\sigma(z) = \frac{e^{z_i}}{\sum_{i=1}^{\mathcal{D}} e^{z_i}}$$

, wobei $z$ die Logits[1] sind. Die Softmax-Funktion ist eine Verallgemeinerung der logistischen Funktion. Sie transformiert die Logits in den Wertebereich $(0, 1]$ und stellt sie als kategoriale Verteilung dar. Die Wahrscheinlichkeitsausgabe der Softmax-Funktion darf aber nicht als Modellunsicherheit interpretiert werden.

[1] Die Ausgabewerte des Netzes bevor die Softmax-Funktion angewendet wird.

## 3.2 Faltendes Neuronales Netzwerk

Wenn ein Neuronales Netzwerk zu groß wird, lässt es sich nur noch schwer oder gar nicht mehr trainieren. Vor allem vollständig verbundene Schichten benötigen viele Parameter. Faltende Neuronale Netzwerke (CNN) wurden ursprünglich für die Bildklassifikation entwickelt und erweitern Neuronale Netze durch sogenannte Faltungsschichten. In der Schicht wird die Faltung zwischen der Eingabe und einem Filter berechnet. Im Vergleich zu einer vollständig verbundenen Schicht, verringert der Filter die Anzahl an Verbindungen zwischen den Neuronen. Es werden nur Eingaben berücksichtigt, die von dem Filter abgedeckt werden. Im Gegensatz zur Bilderkennung, bei der sowohl die Höhe und Breite des Bildes untersucht werden müssen, wird bei einer Zeitreihe der Filter entlang der Zeitachse geschoben. Für einen Zeitpunkt $t$ kann die Faltung einer univaraten Zeitreihe $x$ der Länge $T$ folgendermaßen definiert werden[9]:

$$y_t = \sigma(\langle \omega, x_{t-l/2:t+l/2} \rangle + \beta) | \forall t \in [1, T]$$

$y_t$ ist hier das Ergebnis der Faltung, $\omega$ ist ein Filter der Länge $l$ und $\beta$ ist ein Bias. Abhängig von der Eingabe- und Ausgabedimension können mehrere Faltungen pro Faltungsschicht berechnet werden. Wenn man mehrere Filter auf eine univariate Zeitreihe anwendet, erhält man eine gefilterte multivariate Zeitreihe. Die Schrittlänge des Filters kann dazu genutzt werden die Auflösung, bzw. die Länge der Zeitreihe zu verringern. Kürzere Zeitreihen verringern die Komplexität des Modells, können sich aber bei zu viel Informationsverlust negativ auf die Performanz auswirken.

Batch Normalisierung kann dabei helfen das Netzwerk während des Trainings zu stabilisieren und zu einer besseren Generalisierung führen[16]. Die Operation wird gewöhnlich nach der Faltung auf allen Kanälen der Zeitreihe pro Mini-Batch angewendet:

$$y = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} \odot \alpha + \beta.$$

Auf den extrahierten Features der Faltungsschicht kann eine Pooling Operation ausgeführt werden. Zwei gängige Methoden sind das durchschnittliche Pooling und das maximale Pooling. Dabei berechnet ein sich bewegendes Fenster den größten oder den durchschnittlichen Wert der Eingabe. Die Pooling Schicht verringert die Auflösung der Features und kann eine übermäßige Empfindlichkeit der Faltungsschicht gegenüber dem Standort der Features in der Zeitreihe verringern. Am Ende des Netzwerkes können vollständig verbundene Schichten verwendet werden. Sie verbinden jedes Neuron in einer Schicht mit jedem Neuron einer folgenden Schicht. Dabei werden gewichtete Summen berechnet:

$$y = xA^T + \beta.$$

Es gibt verschiedene Möglichkeiten den Vorhersagewert des Netzwerkes zu interpretieren. Mittels der arg max Funktion erhält man die wahrscheinlichste Klasse, ignoriert dabei aber die restlichen Vorhersagen. Alternativ wird oft die Softmax-Funktion verwendet.

## 3.3 Deep Ensemble

Ein *Ensemble* besteht aus mehreren Modellen, die jeweils eine Vorhersage berechnen. Falls Neuronale Netzwerke verwendet werden, spricht man von einem *Deep-Ensemble*. Beim Training werden die Netzwerke zufällig initialisiert und sind unabhängig voneinander. Sampling Methoden wie *Bagging* (*Bootstrap aggregating*) können die Modelle im Training zusätzlich dekorrelieren. Über eine *Voting* Methode werden mehrere Vorhersagen kombiniert, mit dem Ziel eine besser Vorhersage zu erstellen[8]. Üblicherweise werden dabei die einzelnen Vorhersagen gleichmäßig gewichtet. Die endgültige Vorhersage erhält man, indem der Durchschnitt berechnet wird:

$$p(y|x) = M^{-1} \sum_{m=1}^{M} p_{\theta_m}(y|x).$$

### 3.3.1 Bagging

Bagging erstellt basierend auf einem Trainingsdatensatz $\mathcal{D}$ für jedes der $M$ Modelle einen neuen Datensatz. Die Beispiele werden dabei uniform und mit zurücklegen aus $\mathcal{D}$ gezogen. Wenn die neuen Datensätze $\mathcal{D}_i$ genauso groß sind wie der Ursprüngliche, ist zu erwarten das jeder Datensatz $\mathcal{D}_i$ ungefähr $(1 - e^{-N'/N}) = (1 - \frac{1}{e}) \approx 63.2\%$ einzigartige Beispiele enthält[2].

## 3.4 Monte Carlo Dropout

Dropout ist eine Regularisierungstechnik die von Srivastava et al. [28] entwickelt worden ist. Dabei wird beim Training eine vorher spezifizierte Anzahl, basierend auf der Dropout Rate $p$, von Neuronen in der Dropout Schicht weggelassen und für den kommenden Vorwärtspass nicht berücksichtigt. Der Verlust wird durch Gewichtsanpassungen der übrig gebliebenen Neuronen kompensiert. Hierzu werden ihre Ausgaben um den Faktor $\frac{1}{1-p}$ skaliert. Dropout kann zu einer besseren Generalisierung des Modells führen und verringert die Gefahr einer Überanpassung.

Monte Carlo Dropout[11] ist eine Methode zur Schätzung der Vorhersageunsicherheit in Neuronalen Netzwerken. Dabei verwenden die Netzwerke Dropout nach Faltungsschichten oder vollständig verbundenen Schichten. Im Gegensatz zum normalen Dropout, findet bei der Methode der gleiche Zufallsvorgang auch beim Testen und bei der Vorhersage von neuen Daten statt. Für eine Vorhersage werden mehrere Vorwärtspässe ausgeführt und anschließend der Mittelwert berechnet. Wenn man in der Testphase die Unsicherheit des Modells bestimmt, sollten außerdem alle Eingaben eines Durchgangs die gleichen Neuronen verwenden. Die Dropout-Maske[2] (jeder Dropout Schicht) wird also erst nach dem vollständigen Durchgang neu erstellt. Dadurch stammen alle Modellausgaben eines Durchgangs

[2]Die Dropout-Maske ist eine binäre Matrix.

von der gleichen induzierten A-posteriori-Verteilung ab[10]. Im Standardverfahren soll zusätzlich Ridge Regularisierung (L2) verwendet werden, sie erweitert die Verlustfunktion durch:

$$\text{L2 Verlust} = \text{Verlust} + \lambda \sum_{i}^{N} \omega_i^2, \lambda \geq 0$$

, wobei $\omega$ ein Gewichtsvektor und $\lambda$ die Stärke der Bestrafung ist. Intuitiv werden beim Optimierungsvorgang große Gewichte bestraft, wodurch die Modellkomplexität reduziert wird. Monte Carlo Dropout kann als Ensemble-Modellkombination mit Parameterteilung interpretiert werden[28].

## 3.5 Vorhersageunsicherheit

Die Vorhersageunsicherheit ist das Vertrauen eines Modells in seine Vorhersage. In diesem Kapitel werden zwei unterschiedliche Arten von Unsicherheit vorgestellt. Es wird darauf eingegangen wie ein Modell seine Vorhersagen verbessern und bewerten kann. Zusätzlich werden zwei Funktionen zur Quantifizierung der Vorhersageunsicherheit eingeführt.

### 3.5.1 Aleatorische und Epistemische Unsicherheit

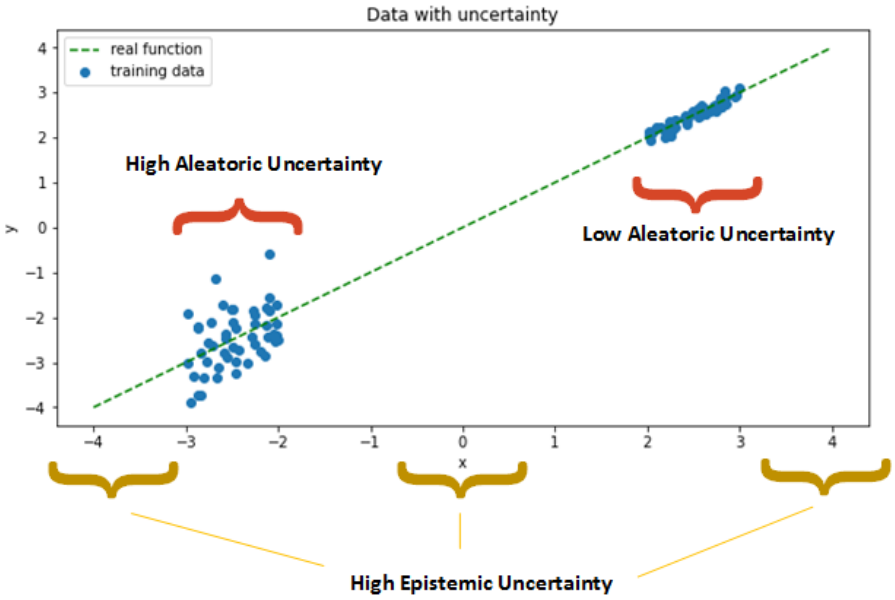



**Abbildung 3.1:** Ein Beispiel für aleatorische und epistemische Unsicherheit in einem linearen Regressionskontext (Quelle: [17]).

Unsicherheit kann aus verschiedenen Quellen stammen. Allgemein unterscheidet man dabei zwischen *aleatorischer* und *epistemischer Unsicherheit*. In der Abbildung 3.1 sind die beiden Unsicherheiten in einem linearen Regressionskontext dargestellt.

**Aleatorische Unsicherheit** ist inhärent in den Daten selbst vorhanden und wird oft auch als Rauschen bezeichnet. In Abbildung 3.1 kann man erkennen, dass die Beispiele mit hoher aleatorischer Unsicherheit mehr von der Regressionsgerade abweichen, als die mit geringer aleatorischer Unsicherheit. Rauschen kann beispielsweise durch ungenaue Messungen oder durch falsche Klassenannahmen entstehen. Aleatorische Unsicherheit kann nicht durch zusätzliche Daten verringert werden, stattdessen kann man sie durch Filter oder Verteilungsannahmen berücksichtigen. Möglicherweise kann auch ein besserer Datenerhebungs- oder Vorverarbeitungsprozess die Unsicherheit verringern.

**Epistemische Unsicherheit** ist auf das Modell zurückzuführen und entsteht durch unzureichendes Wissen. Oft unterscheiden sich die Verteilungen von Test- und Trainingsdaten, sodass eine systematische Abweichung entsteht. Die Trainingsdaten waren dann in ihrem Umfang nicht angemessen. Auch die Modellkonfiguration kann Unsicherheit erzeugen, wie z.B. durch unterschiedliche Architekturen oder Parametereinstellungen. In Abbildung 3.1 kann man erkennen, dass hohe epistemische Unsicherheit in den Bereichen entsteht, die nicht durch die Trainingsdaten abgedeckt wurden. Epistemische Unsicherheit kann durch zusätzliche Daten verringert werden.

### 3.5.2 Kalibrierung und Bewertungs-Regeln

Ein Neuronales Netzwerk sollte neben richtigen Vorhersagen, kalibriertes Vertrauen besitzen. Kalibriertes Vertrauen bedeutet intuitiv, dass die *Genauigkeit*(engl. *accuracy*) und die mit den Vorhersagen assoziierten Wahrscheinlichkeiten möglichst gleich sind. Anzumerken ist hier, dass das Vertrauen nicht basierend auf einer Stichprobe interpretiert werden darf, sondern eine Sammlung von Vorhersagen betrachtet werden muss.
Guo et al.[14] haben eine perfekte Kalibrierung folgendermaßen definiert: Das Klassifikationsproblem wird aus einer probabilistischen Ansicht betrachtet. Die Eingabe $X$ und die Ausgabe $Y$ sind Zufallsvariablen einer multivariaten Verteilung $\pi(X, Y)$. Das neuronale Netzwerk ist eine Funktion $f(X) = (\hat{Y}, \hat{P})$, wobei die Vorhersage aus einem Klassenlabel $\hat{Y}$ und dem damit assoziierten Vertrauen $\hat{P}$ besteht. Eine perfekte Kalibrierung kann definiert werden durch[14]:

$$\mathbb{P}(\hat{Y} = Y|\hat{P} = p) = p, \forall p \in [0, 1]$$

Die Vertrauenswerte sind kalibriert, wenn sie die Wahrscheinlichkeit der Richtigkeit des Netzwerkes genau wiedergeben. In der Praxis, mit nur endlich vielen Stichproben, ist diese Bedingung nicht umsetzbar, da $\hat{P}$ eine kontinuierliche Zufallsvariable ist[14].

Die Kalibierieung eines Neuronalen Netzes kann man beim Training verbessern, indem eine *ordnungsgemäße Bewertungs-Regel* abhängig von der Verlustfunktion maximiert oder minimiert wird. Eine Bewertungs-Regel $S(p_\theta, (y, x)) \rightarrow \mathbb{R}$ bewertet die Qualität einer

Vorhersageverteilung bzgl. einem Ereignis $y|x$ der Grundwahrheitsverteilung $q(y,x)$[19]. Je größer (manchmal auch kleiner) der Wert der Bewertungs-Regel ist, desto besser beschreibt $p_\theta(y|x)$ die wahre Vorhersage $q(y|x)$. Der Erwartungswert ist definiert durch[19]:

$$E(p_\theta, q) = \int q(y,x) S(p_\theta, (y,x)) dy dx.$$

Eine Bewertungs-Regel ist ordnungsgemäß, wenn die beste Qualität durch die wahre Verteilung selbst erreicht wird: also für alle geschätzten Vorhersageverteilungen $E(p_\theta, q) \leq E(q,q)$ gilt. Tatsächlich handelt es sich bei vielen bekannten Verlustfunktionen um ordnungsgemäße Bewertungs-Regeln, wie z.B. dem negativen Log-Likelihood $S(p_\theta, (y,x)) = -\log p_\theta(y|x)$[19]:

$$E(p_\theta, q) = -\mathbb{E}_{q(x)} q(y|x) \log p_\theta(y|x) \geq -\mathbb{E}_{q(x)} q(y|x) \log q(y|x).$$

### 3.5.3 Vorhersageunsicherheit quantifizieren

Die Entropie ist eine Funktion aus der Informationstheorie, sie wurde von Claude Shannon im Jahr 1948 entwickelt[27]. Die Entropie berechnet den durchschnittlichen Informationsgehalt (auch Überraschungs- oder Unsichheitsgehalt) der möglichen Ausgaben einer Zufallsvariable. Aus der Sicht der Informationstheorie, misst die Entropie die Schwierigkeit eine Nachricht zu komprimieren. Je schwieriger die Komprimierung ist, desto mehr Bits sind für die Codierung nötig. Die Anzahl der Bits kann als Informationsgehalt interpretiert werden. Komplexe oder seltene Nachrichten sind schwieriger zu komprimieren und enthalten mehr Informationen (hohe Unsicherheit)[3]. Diese Beziehung kann auf Zufallsexperimente übertragen werden: Man ist überrascht (hoher Informationsgehalt, wenig Vertrauen), wenn ein unwahrscheinliches Ereignis tatsächlich eintritt.
Für eine diskrete Zufallsvariable $X$ ist die Entropie definiert durch:

$$H(X) = -\sum_{x \in X} p_X(x) \log_b p_X(x).$$

Abhängig von der gewählten Basis ist die Ausgabe in Bits $b = 2$, Nats $b = e$ oder Bans $b = 10$. Die Kreuzentropie berechnet die Entropie zwischen zwei Wahrscheinlichkeitsverteilungen über den selben Ereignisraum. Für den diskreten Fall ist sie definiert durch:

$$H(q,p) = -\sum_{x \in X} q_X(x) \log_b p_X(x).$$

Die Kreuzentropie wird oft als Verlustfunktion bei Klassifikationen verwendet. Wenn ein Mehrklassenproblem gegeben ist, ist sie gleich dem negativen Log-Likelihood und somit ebenfalls eine ordnungsgemäße Bewertungs-Regel:

$$H(q,p) = q(y|x) \log \frac{1}{p_\theta(y|x)} = -q(y|x) \log p_\theta(y|x) = -\log p_\theta(y|x).$$

[3] Als einfache, häufig vorkommende Nachrichten $\rightarrow$ (wenig Informationen, niedrige Unsicherheit).

Die *gegenseite Information (eng. mutual information)* berechnet die gegenseitige Abhängigkeit zweier Zufallsvariablen, bzw. die Stärke des statistischen Zusammenhangs. Also wie viel Informationsgehalt man über eine Zufallsvariable erhält, wenn man die jeweils andere Zufallsvariable kennt. Für die beiden diskreten Zufallsvariablen $X, Y$ ist die gegenseitige Information definiert durch:

$$I(X;Y) = -\sum_{y\in Y}\sum_{x\in X} p_{(X,Y)}(x,y)\log\frac{p_{(X,Y)}(x,y)}{p_X(x)p_Y(y)}.$$

Die gegenseitige Information $I(X;Y)$ kann auch durch die Differenz zwischen der Entropie $H(X)$ und der bedingten Entropie $H(X|Y)$ berechnet werden. Die bedingte Entropie enthält die verbleibende Unsicherheit über $X$, wenn man $Y$ kennt. Dementsprechend berechnet die Differenz $H(X) - H(X|Y)$ den Informationsgehalt von $X$ der auch durch $Y$ erklärt wird. Die folgenden Gleichungen sind äquivalent:

$$I(X;Y) \equiv H(X) - H(X|Y) \equiv H(Y) - H(Y|X). \tag{3.1}$$

Die aleatorische und epistemische Unsicherheit wird durch die Vorhersageentropie quantifiziert [11]:

$$H(\hat{y}|x,\mathcal{D}) = -\sum_k p(\hat{y}=k|x,\mathcal{D})\log p(\hat{y}=k|x,\mathcal{D}).$$

$\mathcal{D}$ beschreibt die im Training berücksichtigten Daten, $\hat{y}$ die Modellausgabe und $k$ die verschiedenen Klassen. Die Vorhersageentropie kann über ein Ensemble oder das MC-Dropout Verfahren approximiert werden. Wichtig an dieser Stelle ist, dass beide Verfahren für die gleiche Eingabe mehrere Vorhersagen erstellen. Die individuellen Vorhersagen werden dabei von unterschiedlichen Modellen erzeugt. Die Vorhersageentropie kann approximiert werden durch:

$$H(\hat{y}|x,\mathcal{D}) \approx -\sum_k (\frac{1}{M}\sum_m \sigma(z_m)_k)\log(\frac{1}{M}\sum_m \sigma(z_m)_k) \tag{3.2}$$

,wobei $\{z_m\}_{m=1}^M$ die Logits der unterschiedlichen Modelle sind. $\sigma(z_m)_k$ ist die Softmax-Ausgabe bzgl. der Klasse $k$. Es wird also erst der Mittelwert der Vorhersagen bzgl. einer Klasse über $M$ Modelle berechnet und anschließend die Entropie. Die Vorhersageentropie wird durch zwei Faktoren erhöht. Entweder sind die einzelnen Vorhersagen unsicher, d.h. die Wahrscheinlichkeiten sind nahezu einheitlich, oder die $M$ Vorhersagen sind konträr zueinander.

Die gegenseitige Information zwischen dem A-posteriori über die Modell Parameter $\theta$ und der Vorhersage $\hat{y}$ berechnet nur die epistemische Unsicherheit[11][15]:

$$\begin{aligned} I(\hat{y},\theta|x,\mathcal{D}) &= H(\hat{y}|x,\mathcal{D}) - E_{p(\theta|\mathcal{D})}(\underbrace{H(\hat{y}|x,\theta)}_{Entropie\ \ Likelihood}) \\ &= -\sum_k p(\hat{y}=k|x,\mathcal{D})\log p(\hat{y}=k|x,\mathcal{D}) \\ &\quad + E_{p(\theta|\mathcal{D})}(\sum_k p(\hat{y}=k|x,\theta)\log p(\hat{y}=k|x,\theta)). \end{aligned}$$

Der Erwartungswert des Likelihood der Entropie kann analog als die bedingte Entropie definiert werden und man erhält die bekannte Form der Gleichung 3.1. Die gegenseitige Information kann ebenfalls über ein Ensemble oder das MC-Dropout Verfahren approximiert werden:

$$I(\hat{y}, \theta|x, \mathcal{D}) \approx H(\hat{y}|x, \mathcal{D}) + \frac{1}{M} \sum_m \sum_k \sigma(z_m)_k \log \sigma(z_m)_k.$$

Der erste Term ist die Vorhersageentropie. Ihre Approximation wurde in der Gleichung 3.2 definiert. Der zweite Term ist die erwartete Entropie, bzw. der Mittelwert der Entropie über alle Modellausgaben. Im Gegensatz zur Vorhersageentropie wird nun die Entropie von Vorhersagen, bei denen sich die Modelle im Durchschnitt unsicher sind, ignoriert. Die gegenseitige Information ist hoch, wenn es konträre Vorhersagen mit hohem Vertrauen gibt.

## 3.6 Temperaturskalierung

Guo et al. haben festgestellt, dass moderne Netzwerke schlechter kalibriert sind als ihre Vorgänger[14]. Durch die Verwendung einer ordnungsgemäßen Bewertungsregel als Verlustfunktion wird zwar die Verteilung bzgl. der Trainingsdaten kalibriert, es kann jedoch zu einer Überanpassung kommen. Beispielsweise kann ein Netzwerk an den NLL überangepasst sein, ohne das der gleiche Vorgang beim 0-1-Verlust[4] stattfindet[14].

Temperaturskalierung ist ein Prozess bei dem die Kalibrierung des Netzwerks nachträglich angepasst wird. Sie ist die einfachste Variante der Platt-Skalierung [25]. Ein logistisches Regressionsmodell, bei dem die Logits $z \in \mathbb{R}$ als Features fungieren, soll mittels des Skalars $\mathcal{T} > 0$ kalibrierte Vorhersagen erstellen:

$$y = \sigma(z/\mathcal{T}).$$

Die Genauigkeit der Vorhersagen wird nicht beeinflusst, da alle Logits durch den gleichen Skalar normalisiert werden.

Es gibt zwei Möglichkeiten Temperaturskalierung durchzuführen. Beim ersten Ansatz wird ein Validierungsdatensatz verwendet. Das normale Training findet weiterhin auf den Trainingsdaten statt. Anschließend wird der Skalar auf den Validierungsdaten gelernt. Ein Nachteil dieses Ansatzes ist, dass weniger Daten beim Training zur Verfügung stehen. Beim zweiten Ansatz wird stattdessen ein Teil der Testdaten verwendet[14]. Ein Nachteil ist hier, dass zusätzliche Varianz entsteht. Ashukha et al.[1] reduzieren die Varianz durch 2-fache Kreuzvalidierung während des Testens. Der Testdatensatz wird zufällig in 2 möglichst

[4] Der 0-1-Verlust bestraft alle Entscheidungen, die zu weit von der „richtigen Entscheidung“ entfernt sind, gleich stark und alle restlichen Entscheidungen überhaupt nicht.

gleich große Teilmengen aufgeteilt. Nun werden 2 Durchläufe gestartet, bei denen eine der Teilmengen als Trainingsmenge und die andere Teilmenge als Testmenge verwendet wird[5]. Bei beiden Ansätzen wird $\mathcal{T}$ durch die Minimierung des NLL gelernt. Die restlichen Gewichte des Netzwerkes dürfen bei diesem Vorgang nicht angepasst werden. Je größer der Skalar wird $\mathcal{T} > 1$, desto mehr nähert sich die Wahrscheinlichkeitsausgabe einer Gleichverteilung an. Eine einheitliche Wahrscheinlichkeit ergibt maximale Entropie und damit maximale Unsicherheit. Hingegen wird die Wahrscheinlichkeit bei kleineren Skalaren $\mathcal{T} < 1$ konzentriert.

## 3.7 Das Modell

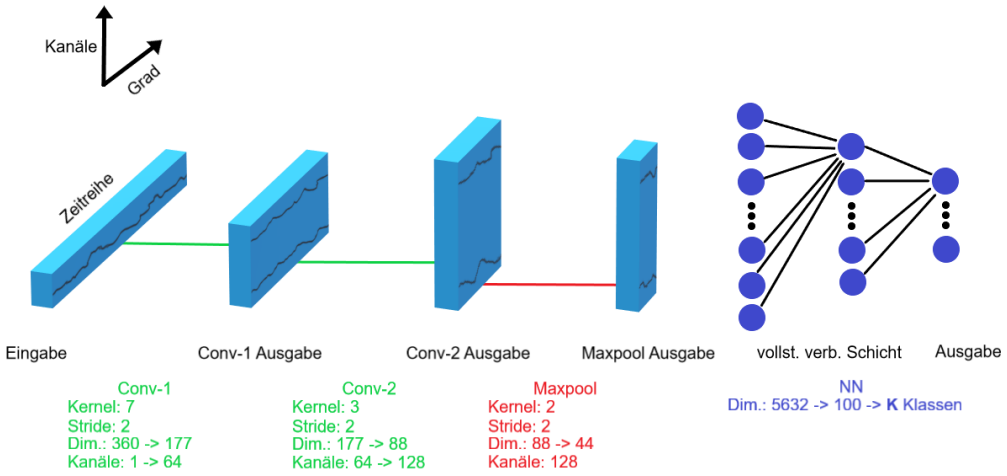



**Abbildung 3.2:** Visualisierung der Modell Architektur.

Es wurden verschiedene Architekturen getestet, dazu gehören u.a. Standard CNN (mit und ohne FC Schicht), ResNet und mehrere parallele CNNs[6]. Ein normales CNN mit einer vergleichsweise breiteren FC Schicht hat die besten Ergebnisse erzielt.

Das entwickelte Modell verwendet die typische Struktur eines normalen CNNs. Nach den Faltungsschichten folgt eine vollständig verbundene Schicht. Da bei einer Zeitreihenanalyse nur eine räumliche Dimension zur Verfügung steht, wird eine eindimensionale Faltungs-, Batch-Normalisierungs und Maxpool Schicht verwendet. In Abbildung 3.2 ist das Modell schematisch visualisiert.

Zunächst versuchen 2 Faltungsschichten die Features der Zeitreihe zu extrahieren. Die Anzahl der Kanäle wird dabei von $1 \rightarrow 64$ und anschließend von $64 \rightarrow 128$ erhöht. Gleichzeitig wird jeweils die räumliche Dimensionalität halbiert. Die beiden Faltungsschichten

[5] Auf der Trainingsmenge wird nur der Skalar $\mathcal{T}$ angepasst, anschließend wird die Testmenge skaliert.
[6] Unterschiedliche Fenstergrößen bei den Faltungen und Konkatenation der Features vor der FC Schicht.

verwenden Batch-Normalisierung und die ReLU Aktivierungsfunktion. Anschließend folgt eine Maxpool Funktion und eine vollständig verbundene Schicht. Letztere verwendet ebenfalls die ReLU Aktivierungsfunktion. Die vollständig verbundene Schicht soll als Puffer zwischen den gelernten Features und der Ausgabe fungieren, mit dem Ziel die extrahierten Features zu interpretieren. Die Ausgabe wird durch die Softmax-Funktion normalisiert und als kategoriale Verteilung dargestellt.

Der MC-Dropout erweitert das oben beschriebene Modell um 2 „sperrbare“ Dropout Schichten. Sie nehmen jeweils Einfluss auf die zweite Faltungsschicht und die vollständig verbundene Schicht. Im Testmodus verwendet die „sperrbare“ Dropout Schicht die gleiche (binäre) Maske für alle Testdaten. Erst wenn der Durchgang abgeschlossen ist, wird eine neue Maske erstellt.

# Kapitel 4

# Datenerhebung

## 4.1 Messrollen Signal Simulator

Der Messrollen Signal Simulator (MRSS) wurde von Preljević et al [7] entwickelt und ist in der Lage realitätsnahe Sensorsignale einer Planheitsmessrolle zu generieren. Die mit dem Simulator erstellten Daten sollten für die Funktion eines Beispieldatensatzes geeignet sein. Die Absicht ist, dass Modelle die auf den Simulatordaten trainiert worden sind, auch auf realen Daten performen. Auf das normale Signal können verschiedene Störsignale addiert werden. Die verfügbaren Störsignale sind charakteristisch für bekannte Fehler. Insgesamt gibt es 4 verschiedene Fehler: weißes Rauschen, Sinusmodulation (Dirty Gab), Tiefpassfilter (Sensorfehler) und Peakmodulation (SIKO-Fehler). Neben dem Simulator wurden 3 verschiedene Konfigurationsdateien bereitgestellt. Sie enthalten Parameterinformationen bezüglich des Bandes, der Rolle, dem Verstärker und der Spule. Falls es nicht explizit erwähnt wird, werden die Standardwerte der Parameter verwendet.

In den folgenden Abschnitten werden die am Simulator vorgenommenen Änderungen beschrieben. Damit der Speicherplatz effizienter genutzt werden kann, werden die Informationen nun in einer MAT Datei der Version 7.3 gespeichert. Die MAT Datei verwendet ein auf HDF5 basiertes Format, wodurch ca. $40-50\%$ weniger Speicherplatz benötigt wird. Damit der Arbeitsspeicher weniger belastet wird und mehrere Matlab Instanzen gleichzeitig für die individuellen Klassen Daten generieren können, werden neue Einträge dynamisch an das jeweilige Dateiende geschrieben. Zusätzlich erstellt jede Instanz eine eigene Datenmatrix, damit beim Ausführen einer neuen Parameterkonfiguration nicht auf Informationen einer anderen Instanz zugegriffen wird.

Die beiden Parameter $T_{Start}$ und $T_{Max}$ spezifizieren die Laufzeit des Simulators. $T_{Start}$ legt den Startzeitpunkt der Simulation fest, während $T_{Max}$ die Dauer derselbigen spezifiziert. Der Simulator generiert, abhängig von der spezifizierten Dauer, unterschiedlich viele Einträge und speichert sie in einer Tabelle. Tabelleneinträge werden pro Grad Umdrehung erstellt, dementsprechend werden bei einer vollständigen Rotation 360 Einträge

| Sensor | 1 2 | 3 4 | 5 6 | 7 8 | 9 10 | 11 12 | 13 14 | 15 16 | 17 18 |
|---|---|---|---|---|---|---|---|---|---|
| Channel | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

| Sensor | 19 20 | 21 22 | 23 24 | 25 26 | 27 28 | 29 30 | 31 32 | 33 34 | 35 36 |
|---|---|---|---|---|---|---|---|---|---|
| Channel | 10 | 11 | 12 | 10 | 13 | 14 | 15 | 16 | 17 |

| Sensor | 37 38 | 39 40 | 41 42 | 43 44 | 45 46 | 47 48 | 49 50 |
|---|---|---|---|---|---|---|---|
| Channel | 18 | 19 | 20 | 21 | 22 | 23 | 24 |

**Tabelle 4.1:** Sensor-Kanal Aufteilung.

generiert. Da ein Beispiel aus einer kompletten Rotation bestehen soll, werden nun unvollständige Rotationen am Tabellenanfang und -ende gelöscht. Damit die Einträge dem Modell einfacher übergeben werden können, wird die Tabelle in eine dreidimensionale Matrix transformiert. Die Matrix besitzt die Achsenunterteilung (`Grad`, `Kanal`, `Beispiel`), damit beim Einlesen in ein Array, die für den späteren Anwendungszweck nötige Unterteilung (`Beispiel`, `Kanal`, `Grad`) angenommen wird. In seiner Standardversion ist der Simulator nur in der Lage eine Parameterkonfiguration pro Ausführung zu berücksichtigen. Damit mehrere Parameterkombinationen berücksichtigt werden und ein möglichst vollständiger Datensatz generiert werden kann, wurde zusätzlich eine Gridsuche implementiert.

## 4.2 Planheitsmessrollen- und Signaleigenschaften

Der Klassifikator soll zwischen den folgenden 5 Klassen unterscheiden: normales Signal, weißes Rauschen, Dirty Gap, Sensorfehler und SIKO-Fehler. Die einzelnen Klassen und ihre Eigenschaften werden in den folgenden Kapiteln beschrieben. Zunächst werden verschiedene Eigenschaften der Planheitsmessrolle und des Signals vorgestellt.

Der Beispieldatensatz wird mithilfe des Messrollen Signal Simulators erstellt.Die simulierte Planheitsmessrolle besitzt insgesamt 50 Sensoren. Der Simulator generiert pro Sensor ein Signal, die Signallänge hängt von der spezifizierten Laufzeit $T_{Max}$ ab. Ein Signal beschreibt die Stromspannung pro Grad Umdrehung. Die Signale mehrere Sensoren werden durch verschiedene Filter angepasst und in Kanälen zusammengefasst. Die betrachtete Planheitsmessrolle verwendet insgesamt 24 Kanäle. Mit Ausnahme von Kanal 10, welcher die Informationen von 4 Sensoren enthält, beinhalten die restlichen Kanäle Informationen von jeweils 2 Sensoren. Die Sensor-Kanal-Aufteilung ist in Tabelle 4.1 veranschaulicht. Die Sensoren 1 und 50 befinden sich jeweils an den äußeren Enden der Messrolle. Zur Analyse und Klassifikation der Fehlerszenarien werden die Kanalsignale[1] verwendet. Sie können als Zeitreihen interpretiert werden. Damit die Signale besser analysiert und den

[1] Kanalsignal: Das in dem Kanal gefilterte und zusammengefasste Signal mehrerer Sensoren.

Klassen zugeordnet werden können, werden sie in kleinere Sequenzen aufgeteilt. Bei der Aufteilung wurde eine Schrittlänge von 360 Grad gewählt. Die Zeitreihen enthalten dann die Informationen einer kompletten Rotation.

Die Planheitsmessrolle besitzt eine feste Breite, während die aufgelegte Bandbreite $w_{Band}$ variieren kann. Ist das aufgelegte Band mindestens so breit wie die Planheitsmessrolle selbst, sind alle Sensoren überdeckt. Wenn das Band kleiner ist, können außenliegende Sensoren gar nicht oder nur teilweise überdeckt sein. In Abbildung 4.1 ist eine Voll- und Teilüberdeckung schematisch dargestellt.

**(a)** Vollüberdeckung **(b)** Teilüberdeckung

**Abbildung 4.1:** Visualisierung einer Voll- und Teilüberdeckung des Bandes (grau schraffierte Fläche) auf der Planheitsmessrolle.

Wenn die Sensoren vollständig von einem Band überdeckt sind, ähneln sich die Kanalsignale sehr. In Abbildung 4.2 kann man erkennen, dass das Signal von Sensoren die näher an der Messrollenkante liegen, eine etwas kleinere Amplitude erzeugen. Falls ein Beispiel

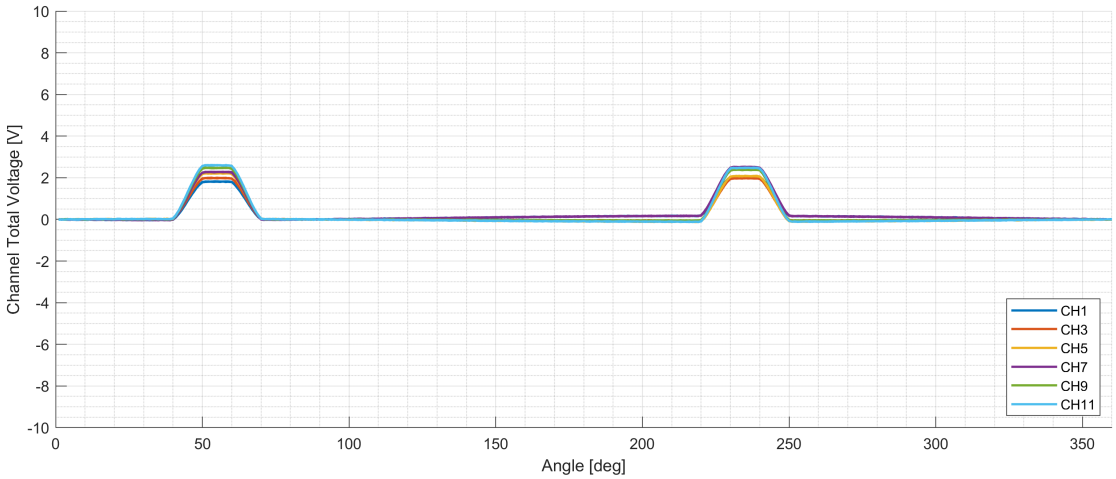



**Abbildung 4.2:** Kanäle 1,3,5,7,9 und 11 bei einer Vollüberdeckung ($N.Power = 0.01V$, $\varphi_{entry} = 0$ und $\varphi_{exit} = 20$).

die Zeitreihen aller 24 Kanäle enthält, würde es viele redundante Informationen enthalten. Beispielsweise beeinflusst ein Sensorfehler nur einen Kanal. Ein Beispiel würde dann 23·360 redundante Datenwerte enthalten. Die anderen Fehler ähneln sich Kanal übergreifend. Aufgrund des mangelnden Informationsgewinns und den hohen Kapazitätsanforderungen einer multivariaten Zeitreihe mit 24 Dimensionen, werden die Signale einzeln betrachtet. Ein

Beispiel enthält dann die Zeitreihe eines Kanals. In der realen Anwendung müssten dann, abhängig von der Bandbreite, mehrere Modelle parallel die einzelnen Kanäle klassifizieren. Die Ausgaben könnten anschließend durch eine Aggregationsmethode interpretiert werden.

Insgesamt kann man über alle Kanäle 3 sich wiederholende Grundstrukturen erkennen. Sie sind in Abbildung 4.3 visualisiert. Der Kanal 10 fasst die Informationen von 4 Sensoren zusammen, weshalb das Signal dieselbige Anzahl an Amplituden enthält. Die anderen Kanäle folgen einer ähnlichen Struktur wie Kanal 11 oder 12 und man kann sie in die Gruppen A[2] und B[3] unterteilen. Ein Modell das auf Beispielen von den Kanälen 10,11 und 12 trainiert worden ist und somit die 3 Grundstrukturen lernen kann, generalisiert auf die restlichen Kanäle und erzielt auf ihnen vergleichbare Ergebnisse. Die Performanz kann leicht verbessert werden, wenn zusätzlich Beispiele basierend auf den Kanälen 1 und 2 verwendet werden. Sie enthalten Signale mit den vergleichsweise kleinsten Amplituden (Abb. 4.2). Damit die 3 Grundstrukturen und die leicht variierenden Amplitudengrößen im Datensatz enthalten sind, werden Beispiele basierend auf den Kanäle 1, 2, 10, 11 und 12 verwendet.

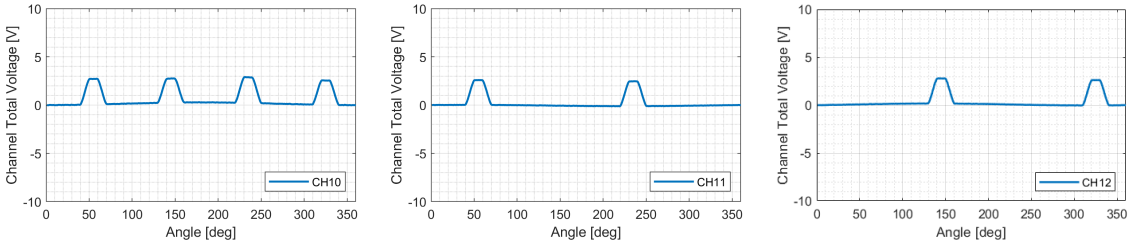
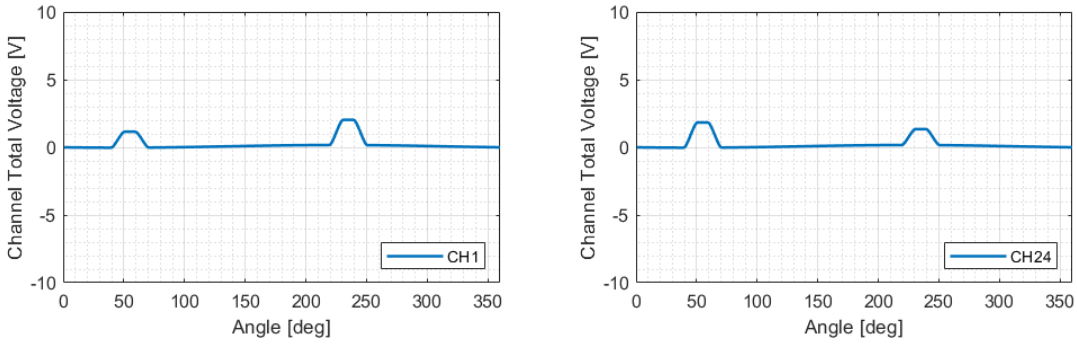



**Abbildung 4.3:** Beispiel für die 3 Grundstrukturen eines Signals bei einer Vollüberdeckung ($N.Power = 0.01V$, $\varphi_{entry} = 0$, $\varphi_{exit} = 20$).

Falls ein Sensor nur teilweise von einem Band überdeckt ist, verkleinert sich die zugehörige Amplitude im Signal. In Abbildung 4.4 ist ein Beispiel visualisiert. Dieser Vorgang beeinflusst nur Kanäle die Bandkanten analysieren. Fälle der Teilüberdeckung werden separat von der Vollüberdeckung durch ein eigenständiges Modell behandelt. Auch hier werden die Kanalsignale einzeln analysiert. Es kann davon ausgegangen werden, dass im Normalbetrieb die in der Mitte liegenden Sensoren der Messrolle vollständig überdeckt sind. Dementsprechend müssen nur 2 der Grundstrukturen berücksichtigt werden, bzw. die Struktur von Kanal 11 und 12 (Abb. 4.3). Abhängig davon ob ein Sensor näher an der linken oder der rechten Messrollenkante liegt, wird das Kanalsignal bei einer Teilüberdeckung unterschiedlich beeinflusst. Angenommen das Band wird kontinuierlich schmaler und beide Sensoren sind anfänglich überdeckt:

[2] $A = \{1, 3, 5, 7, 9, 11, 14, 16, 18, 20, 22, 24\}$
[3] $B = \{2, 4, 6, 8, 12, 13, 15, 17, 19, 21, 23\}$




**(a)** Kanal 1 **(b)** Kanal 24

**Abbildung 4.4:** Teilüberdeckung im Kanal 1 und 24 ($Bandbreite = 1690$, $\varphi_{entry} = 0$, $\varphi_{exit} = 20$).

- Wenn die Sensoren näher an der **linken** Messrollenkante liegen, wird zunächst die linke Amplitude beeinflusst. Erst wenn diese verschwunden ist[4], würde die rechte Amplitude ab einer bestimmten Bandbreite kleiner werden (Abb. 4.4a).

- Wenn die Sensoren näher an der **rechten** Messrollenkante liegen, findet ein ähnlicher Vorgang statt. Hier würde als erstes die rechte Amplitude kleiner werden(Abb. 4.4b).

Für das Modell der Teilüberdeckung werden Beispieldaten basierend auf den Kanälen 1, 2, 23 und 24 erstellt. Zusätzlich werden unterschiedliche Bandbreiten berücksichtigt, damit alle charakteristischen Eigenschaften einer Teilüberdeckung im Datensatz enthalten sind.

## 4.3 Grundkonfiguration

Die Grundkonfiguration beinhaltet Parameter die sowohl für den Normalfall, als auch für die Fehlerklassen relevant sind. Falls es Änderungen bezüglich dieser Parameter gibt, wird es explizit in folgenden Kapiteln erwähnt. In der Tabelle 4.2 wird eine Übersicht gegeben.

Die Rotationsgeschwindigkeit der Planheitsmessrolle beschleunigt am Anfang und Ende der möglichen Simulationszeit, wodurch sich das Signal vom Normalverlauf unterscheidet. Dieser Zustand ist verhältnismäßig kurz und wird ignoriert. Der Startzeitpunkt wurde auf $T_{Start} = 1500$ festgelegt. Die Rotationsgeschwindigkeit der Planheitsmessrolle variiert hier nur noch wenig und befindet sich in einem stationären Zustand.

Die simulierte Planheitsmessrolle besitzt eine Messflächenbreite von $1720mm$. Für den Fall einer Vollüberdeckung wird eine Bandbreite von $w_{Band} = 1720mm$ verwendet, d.h. alle Sensoren sind vollständig vom Band überdeckt. Bei einer Teilüberdeckung müssen unterschiedliche Bandbreiten berücksichtigt werden. Für die Kanäle 1 und 24 werden Bandbreiten im Bereich von $w1_{Band} = 1615 - 1655mm$ und $w2_{Band} = 1670 - 1720mm$ verwendet. Für die Kanäle 2 und 23 werden Bandbreiten im Bereich von $w1_{Band} = 1515 - 1555mm$ und

[4] D.h. der Sensor ist nicht mehr überdeckt.

$w2_{Band} = 1570 - 1620mm$ verwendet. Die in $w2_{Band}$ spezifizierten Breiten beeinflussen die jeweils „außen liegende Amplitude“[5], während $w1_{Band}$ die „innen liegenden Amplituden“[6] beeinflusst.

Im normalen Betrieb kann es vorkommen, dass das Band nicht exakt mittig über die Planheitsmessrolle läuft. Die axialen Verschiebungen können zu asymmetrischen Verhalten des Bandes an den Bandkanten führen. Im Simulator kann über den Parameter $B_{Misalign}$ die axiale Verschiebung eingestellt werden. Dieser Vorgang wird bereits durch das einzelne Analysieren der Kanäle mit unterschiedlichen Bandbreiten simuliert. Der Parameter $B_{Misalign}$ wurde auf dem Standartwert von $0mm$ gelassen.

Der Eingangswinkel $\phi_{entry}$ und der Ausgangswinkel $\phi_{exit}$ des Bandes auf der Planheitsmessrolle kann jeweils zwischen $1 - 10°$ und $10 - 30°$ Grad variieren. In den Beispieldaten werden alle Grad Kombinationen berücksichtigt. Nur Fälle bei denen der Eingangs- und der Ausgangswinkel gleich sind werden ignoriert. Hier ist keine Amplitude im Signal vorhanden.

Das Signal kann durch ein Grundrauschen verzerrt werden. Der Parameter $N_{Power}$ spezifiziert die Intensität des Rauschens. Normalerweise beträgt die Rauschstärke ca. $20mV$, weshalb auf die generierten Beispiele Stärken von 0 bis $30mV$ angewendet werden.

Wenn die Geberspalten mit Staub verstopft sind[7], kann sich eine Sinusmodulation auf dem Signal bilden. Der Parameter $DG_{Amplitude}$ beschreibt die Schwingungsamplitude der Sinusoide und damit den Einfluss der Störung auf das Signal. Sinusmodulationen zwischen 0 und $30mV$ sollen keinen Fehler ausgeben und werden in der Grundkonfiguration berücksichtigt.

| | **Vollüberdeckung** | | | **Teilüberdeckung** | | |
|---|---|---|---|---|---|---|
| | Von/Bis | Schritt | Anzahl | Von/Bis | Schritt | Anzahl |
| $w_{Band}$ $[mm]$ | 1720 | | 1 | 1515 - 1555 | 5 | 9 |
| | | | | 1570 - 1620 | 5 | 11 |
| | | | | 1615 - 1655 | 5 | 9 |
| | | | | 1670 - 1720 | 5 | 11 |
| $\phi_{Entry}$ [°] | 0 - 10 | 1 | 11 | 0 - 10 | 1 | 11 |
| $\phi_{Exit}$ [°] | 10 - 30 | 1 | 21 | 10 - 30 | 1 | 21 |
| $N_{Power}[mV]$ | 0 - 30 | 10 | 4 | 0 - 30 | 10 | 4 |
| $DG_{Amplitude}[mV]$ | 0, 30 | | 2 | 0, 30 | | 2 |

**Tabelle 4.2:** Übersicht der Grundkonfiguration.

Die Anzahl an möglichen Parameterkombinationen $PK$ für die unterschiedlichen Klassen variiert zum Teil stark. Für kleinere, unterrepräsentierte Klassen werden deshalb mehr

[5] „außen liegende Amplitude“ → Amplitude der Sensoren, die näher an der Messrollenkante liegen.
[6] „innen liegenden Amplituden“ → Amplitude der Sensoren, die näher an der Messrollenmitte liegen.

Beispiele generiert. Die Variable $nSamples$ spezifiziert die Anzahl an Beispielen pro Parameterkombination. Die Datensatzgröße $DS$ wird berechnet durch:

$$DS = PK \cdot |Kanäle| \cdot nSamples.$$

**Datensatzgröße der fehlerfreien Klasse:**

$$\begin{aligned} PK &= |w_{Band}| \cdot |\phi_{entry}| \cdot |\phi_{exit}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \\ &\quad - |w_{Band}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \\ PK_{Voll} &= 1 \cdot 11 \cdot 21 \cdot 4 \cdot 2 - 1 \cdot 4 \cdot 2 = 1840 \\ DS_{Voll} &= 1840 \cdot 5 \cdot 360 = 3312000 \\ PK_{Teil} &= 20 \cdot 11 \cdot 21 \cdot 4 \cdot 2 - 20 \cdot 4 \cdot 2 = 36800 \\ DS_{Teil} &= 36800 \cdot 4 \cdot 22 = 3238400 \end{aligned}$$

Der Subtrahend in $PK$ beinhaltet jeweils die Anzahl der Parameterkombinationen die ignoriert werden, weil der Eingangs- und der Ausgangswinkel gleich sind.

Bei der Erstellung der normalen Klasse wird ausschließlich die Grundkonfiguration verwendet. Eine Übersicht ist in Tabelle 4.2 gegeben. Beispielsweise gibt es bei der Vollüberdeckung 1840 Parameterkombinationen, es werden 5 Kanäle[7] berücksichtigt und pro Kombination werden 360 Beispiele generiert. Die Datensatzgröße des fehlerfreien Falls beträgt für die Vollüberdeckung 3.312.000 und für die Teilüberdeckung 3.238.400 Beispiele.

## 4.4 Fehlerklassenkonfiguration

Die Fehlerklassenkonfiguration passt die Grundkonfiguration an die jeweilige Fehlerklasse an. In der Tabelle 4.3 sind die Wertebereiche und die gewählten Schrittgrößen zusammengefasst.

### 4.4.1 Weißes Rauschen

Jedes Signal einer Planheitsmessrolle enthält ein Grundrauschen von ca. 20mV. Steigt das Rauschen, kann dies auf mögliche Fehlerquellen wie bspw. Lagerschäden hinweisen[7]. Der kritische Zustand wird erreicht, wenn die Amplitude des Rauschens über 40mV ansteigt. Im Datensatz wurde für den Fehlerfall Rauschstärken zwischen $40 - 80mV$ berücksichtigt.

[7] Kanäle: 1, 2, 10, 11 und 12.

| | Voll- und Teilüberdeckung | | |
|---|---|---|---|
| | Von/Bis | Schritt | Anzahl |
| **Weißes Rauschen** | | | |
| $N_{Power}[mV]$ | 40 - 80 | 10 | 5 |
| **Sinusmodulation** | | | |
| $DG_{Amplitude}[mV]$ | 40 - 80 | 10 | 5 |
| **Tiefpassfilter** | | | |
| $SE_{Frequenz}[Hz]$ | 1 - 30 | 2 | 15 |

| | Vollüberdeckung | | |
|---|---|---|---|
| | Von/Bis | Schritt | Anzahl |
| **SIKO-Fehler** | | | |
| $SIKO_{Degree}[°]$ | 1 - 360 | 1 | 360 |

**Tabelle 4.3:** Übersicht der Fehlerkonfiguration.

### Datensatzgröße der Fehlerklasse „weißes Rauschen“:

$$
\begin{aligned}
PK &= |w_{Band}| \cdot |\phi_{entry}| \cdot |\phi_{exit}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \\
&\quad - |w_{Band}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \\
PK_{Voll} &= 1 \cdot 11 \cdot 21 \cdot 5 \cdot 2 - 1 \cdot 5 \cdot 2 = 2300 \\
DS_{Voll} &= 2300 \cdot 5 \cdot 288 = 3312000 \\
PK_{Teil} &= 20 \cdot 11 \cdot 21 \cdot 5 \cdot 2 - 20 \cdot 5 \cdot 2 = 46000 \\
DS_{Teil} &= 46000 \cdot 4 \cdot 18 = 3312000
\end{aligned}
$$

Die Datensatzgröße der Klasse „weißes Rauschen“ beträgt für die Vollüberdeckung und Teilüberdeckung 3.312.000 Beispiele. In Abbildung 4.5a ist ein fehlerbehaftetes Signal mit einer Rauschstärke von $40mV$ dargestellt.

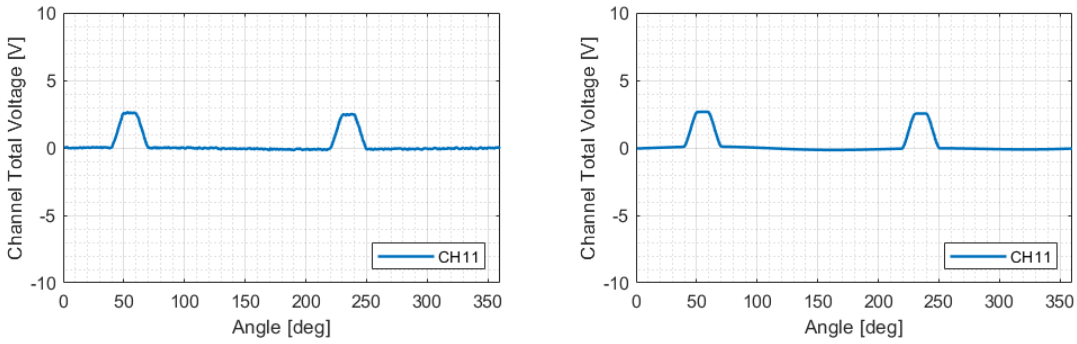
### 4.4.2 Sinusmodulation

Mit der Zeit können die Geberspalten mit Staub verstopfen und eine Sinusmodulation kann sich auf dem Signal bilden[7]. Bei einer Sinusmodulation von $40mV$ soll ein Fehler ausgegeben werden. Im Datensatz wurden Schwingungsamplituden zwischen $40 - 80mV$ berücksichtigt.

**Datensatzgröße der Fehlerklasse „Dirty Gap“:**

$$
\begin{aligned}
PK &= |w_{Band}| \cdot |\phi_{entry}| \cdot |\phi_{exit}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \\
&\quad - |w_{Band}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \\
PK_{Voll} &= 1 \cdot 11 \cdot 21 \cdot 4 \cdot 5 - 1 \cdot 4 \cdot 5 = 4600 \\
DS_{Voll} &= 4600 \cdot 5 \cdot 72 = 3312000 \\
PK_{Teil} &= 20 \cdot 11 \cdot 21 \cdot 4 \cdot 5 - 20 \cdot 4 \cdot 5 = 92000 \\
DS_{Teil} &= 92000 \cdot 4 \cdot 9 = 3312000
\end{aligned}
$$

Die Datensatzgröße der Klasse „Dirty Gap“ beträgt für die Vollüberdeckung und die Teilüberdeckung 3.312.000 Beispiele. In Abbildung 4.5b ist ein fehlerbehaftetes Signal mit einer Schwingungsamplitude von $40mV$ dargestellt.




**(a)** Weißes Rauschen ($N_{Power} = 40mV$) **(b)** Dirty Gap ($DG_{Amplitude} = 40mV$)

**Abbildung 4.5:** Beispiel für die Fehlerszenarien weißes Rauschen und Dirty Gap ($\varphi_{entry} = 0$, $\varphi_{exit} = 20$).

### 4.4.3 Tiefpassfilter

Falls ein Sensor defekt ist, kann sich das Signal deformieren und die Amplitude sinkt unter der Belastung des Bandes. Die Deformation ähnelt einem Signal, welches mit einem Tiefpassfilter gefiltert worden ist[7]. In der Simulation wurde eine Butterworth-Filterordnung von 2 benutzt und die Grenzfrequenz wird von 30 auf $1mV$ verringert. Je kleiner die Grenzfrequenz wird, desto größer ist die Amplitudenreduzierung. Der Parameter $SE_{Sensor}$ spezifiziert die defekten Sensoren. Es wird davon ausgegangen, dass maximal ein Sensor pro Kanal ausfallen kann. Für den Kanal 10 müssen 4 Fälle und für die restlichen Kanäle müssen 2 Fälle berücksichtigt werden.

**Datensatzgröße der Fehlerklasse „Sensorfehler“:**

$$\begin{aligned} PK &= |w_{Band}| \cdot |\phi_{entry}| \cdot |\phi_{exit}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \cdot |SE_{Frequenz}| \cdot |SE_{Sensor}| \\ &\quad - |w_{Band}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \cdot |SE_{Frequenz}| \cdot |SE_{Sensor}| \\ PK1_{Voll} &= 1 \cdot 11 \cdot 21 \cdot 4 \cdot 2 \cdot 15 \cdot 2 - 1 \cdot 4 \cdot 2 \cdot 15 \cdot 2 = 55200 \ (Kanal\ mit\ 2\ Sensoren) \\ PK2_{Voll} &= 1 \cdot 11 \cdot 21 \cdot 4 \cdot 2 \cdot 15 \cdot 4 - 1 \cdot 4 \cdot 2 \cdot 15 \cdot 4 = 110400 \ (Kanal\ mit\ 4\ Sensoren) \\ DS_{Voll} &= 55200 \cdot 4 \cdot 10 + 110400 \cdot 1 \cdot 10 = 3312000 \\ PK_{Teil} &= 20 \cdot 11 \cdot 21 \cdot 4 \cdot 5 - 20 \cdot 4 \cdot 5 = 92000 \\ DS_{Teil} &= 92000 \cdot 4 \cdot 1 = 3312000 \end{aligned}$$

Die Datensatzgröße der Klasse „Sensorfehler“ beträgt für die Vollüberdeckung 3.974.400 und die Teilüberdeckung 3.312.000 Beispiele. In Abbildung 4.5b ist ein fehlerbehaftetes Signal mit einer Grenzfrequenz von $15Hz$ dargestellt.

### 4.4.4 Peakmodulation

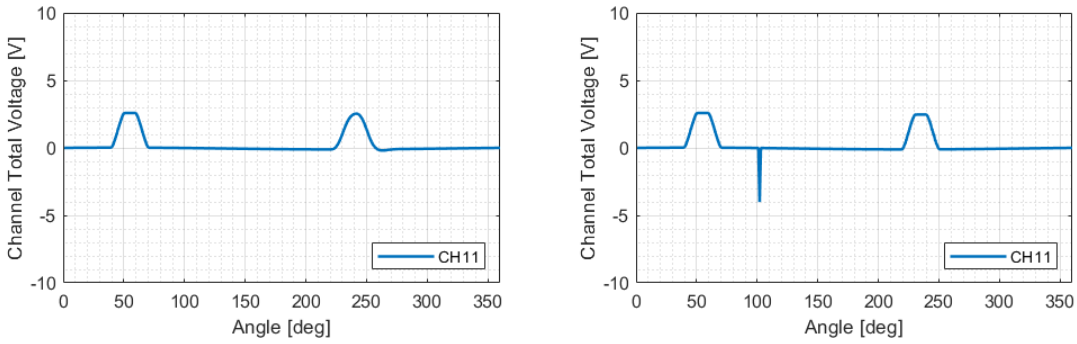
Ist ein Elektronikbauteil wie der SIKO defekt, zeigen sich im Signal Übertragungsfehler. In der Regel wird zunächst ein Datenpunkt alle $n$-Umdrehungen nicht gesendet oder verzerrt. Die Häufigkeit des Übertragungsfehlers steigt mit der Zeit, bis in jeder Rollenumdrehung mehrere Fehler auftreten[7]. Bereits ein Übertragungsfehler soll als Schwellwert für die Auslösung eines Fehlers angesehen werden. Der Fehler kann an einer beliebigen Stelle im Signal auftreten und wird auf jedem Kanal ausgelöst. Im Datensatz wurden Beispiele für alle 360 möglichen Positionen ($SIKO_{Degree}$) berücksichtigt. Dabei wird die Stärke der Peaks zufällig zwischen 1 und $4V$ variiert.

Bei der Teilüberdeckung ist durch die große Anzahl an Parameterkombinationen die Berücksichtigung des Fehlerfalls mit einer Gridsuche zu komplex. Der resultierende Datensatz wäre zu groß und die Simulation, in ihrer aktuellen Implementierung, würde zu viel Zeit in Anspruch nehmen. Da der Peak in jedem Kanal gleichzeitig und an derselben Position erscheint, wird der Fehlerfall aber bereits durch die Vollüberdeckung behandelt.

**Datensatzgröße der Fehlerklasse „SIKO-Fehler“:**

$$\begin{aligned} PK &= |w_{Band}| \cdot |\phi_{entry}| \cdot |\phi_{exit}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \cdot |SIKO_{Degree}| \\ &\quad - |w_{Band}| \cdot |N_{Power}| \cdot |DG_{Amplitude}| \cdot |SIKO_{Degree}| \\ PK_{Voll} &= 1 \cdot 11 \cdot 21 \cdot 4 \cdot 2 \cdot 360 - 1 \cdot 4 \cdot 2 \cdot 360 = 662400 \\ DS_{Voll} &= 662400 \cdot 5 \cdot 1 = 3312000 \end{aligned}$$

Die Datensatzgröße der Klasse „SIKO-Fehler“ beträgt für die Vollüberdeckung 3.312.000 Beispiele. In Abbildung 4.6b ist ein fehlerbehaftetes Signal mit einer Amplitude von $4V$ an der Position 101° dargestellt.




**(a)** Sensorfehler ($SE_{Frequenz} = 15Hz$) **(b)** SIKO-Fehler ($SIKO_{Degree} = 101°$)

**Abbildung 4.6:** Beispiel für die Fehlerszenarien Sensorfehler und SIKO-Fehler ($\varphi_{entry} = 0$, $\varphi_{exit} = 20$).

# Kapitel 5

# Metriken

## 5.1 Definitionen

**Genauigkeit (ACC)**⇑[1] ist eine Möglichkeit Klassifikatoren zu evaluieren. Sie berechnet den Anteil der richtigen Vorhersagen und ist definiert durch:

$$\mathrm{ACC} = \frac{1}{N}\sum_{n=1}^{N} \chi_T(\hat{y}_n, y_n).$$

Die Indikatorfunktion ist definiert durch:

$$\chi_T(a,b) = \begin{cases} 1, & \text{falls } a = b, \\ 0, & \text{falls } a \neq b. \end{cases}$$

**Durchschnittlicher negativer Log-Likelihood (NLL)**⇓ ist eine ordnungsgemäße Bewertungsregel und kann zum evaluieren der Modellunsicherheit verwendet werden. Der durchschnittliche negative Log-Likelihood ist definiert durch:

$$\mathrm{NLL} = \frac{1}{N}\sum_{n=1}^{N} \log p(y = y_n | x_n),$$

Je mehr sich die Vorhersageverteilung $p(y|x)$ und die Grundwahrheitsverteilung $q(y|x)$ ähneln, desto kleiner ist der NLL. Intuitiv ist der NLL groß, wenn kleine Wahrscheinlichkeiten für das richtige Klassenlabel und große Wahrscheinlichkeiten für die falschen Klassenlabel vorhergesagt werden.

**Brier Score (BS)**⇓ [5] ist eine ordnungsgemäße Bewertungsregel und misst die Genauigkeit von Vorhersagewahrscheinlichkeiten. Der Brier Score ist definiert durch:

$$\mathrm{BS} = \frac{1}{N}\frac{1}{K}\sum_{n=1}^{N}\sum_{k=1}^{K}(p(y = k|x_n) - \chi_T(k, y_n))^2.$$

[1] ⇓ kleinere / ⇑ größere Werte sind besser

Die Bewertungsregel lässt sich in 3 Teile zerlegen[20]: BS = UNC − RES + REL. Die Uncertainty (UNC) ist die inhärente Unsicherheit des Ausgangs eines Ereignisses, bzw. die marginale Unsicherheit der Klassenlabel. Die Resolution (RES) misst wie groß der Unterschied zwischen den Vorhersagen und dem durchschnittlichen Vorhersagewert ist. Die Reliability (REL) berechnet die Kalibrierung.

**Expected Calibration Error (ECE)**⇓ [21] kann zur Schätzung der Fehlkalibrierung verwendet werden. Zunächst werden die Vorhersagen basierend auf ihren Vorhersagewahrscheinlichkeiten und den Grenzwerten $\rho$ in Klassen(Bins) eingeteilt. Anschließend wird der durchschnittliche Unterschied zwischen den Genauigkeiten und den Vorhersagewahrscheinlichkeiten innerhalb der $M$ Klassen, über insgesamt $N$ Beispiele gemessen, $B_m = \{n \in [1, N] : p(y = \hat{y}_n|x_n) \in (\rho_m, \rho_{m+1}]\}$. Der ECE ist definiert durch:

$$\begin{aligned}
\text{ECE} &= \sum_{m}^{M} \frac{|B_m|}{N} |\text{acc}(B_m) - \text{conf}(B_m)| \\
\text{acc}(B_m) &= |B_m|^{-1} \sum_{n \in B_m} \chi_T(\hat{y}_n, y_n), \\
\text{conf}(B_m) &= |B_m|^{-1} \sum_{n \in B_m} p(y = \hat{y}_n|x_n), \\
\hat{y}_n &= \arg\max_y p(y|x_n).
\end{aligned}$$

**Thresholded Adaptive Calibration Error (TACE)**⇓ [23] ist eine Erweiterung des ECE. Im Gegensatz zum ECE, der nur die maximale Vorhersagewahrscheinlichkeit berücksichtigt, werden nun alle Vorhersagewahrscheinlichkeiten verwendet. Sehr kleine Vorhersagewahrscheinlichkeiten können dazu führen, dass die Klassen nicht effizient genutzt werden und damit die Kalibrierungsbewertung negativ beeinflussen. Ein Grenzwert (engl. threshold) sorgt dafür, dass kleinere Wahrscheinlichkeiten ignoriert werden. Der ECE besitzt einen Bias-Varianz-Tradeoff, dieser wird im TACE durch adaptive Klassenreichweiten behoben. Jede Klasse enthält dann die gleiche Anzahl an Vorhersagen. Die Vorhersagen eines Beispiels werden, abhängig von seinen Vorhersagewahrscheinlichkeiten und dem Grenzwert $\gamma$, den Klassen zugeordnet, $B_m^{(A)} = \{n \in [1, N], k \in [1, K] : (p(y_k|x_n) > \gamma) \in (\rho_m^{(A)}, \rho_{m+1}^{(A)}]\}$. Der TACE ist definiert durch:

$$\begin{aligned}
\text{TACE} &= \frac{1}{K}\frac{1}{M} \sum_{k}^{K} \sum_{m}^{M} \frac{|B_m^{(A)}|}{N} |\text{acc}(B_m^{(A)}, k) - \text{conf}(B_m^{(A)}, k)|, \\
\text{acc}(B_m^{(A)}, k) &= |B_m^{(A)}|^{-1} \sum_{n \in B_m^{(A)}} \chi_T(k, y_n), \\
\text{conf}(B_m^{(A)}, k) &= |B_m^{(A)}|^{-1} \sum_{n \in B_m^{(A)}} p(y = k|x_n).
\end{aligned}$$

**Komplett Out-Of-Domain Fälle**: OOD Daten stammen nicht aus der gelernten Verteilung und besitzen kein Klassenlabel. Wenn die Modellreaktion bei solchen Fällen getestet werden soll, kann stattdessen das Vertrauen, die Vorhersageentropie und die gegenseitige Information untersucht werden (siehe Kapitel 3.5.3). Mehr Informationen werden im Kapitel 6 bei dem jeweiligen OOD Experiment gegeben.

## 5.2 Schwächen

Die vorgestellten Metriken haben verschiedene Schwächen. Der NLL kann Randwahrscheinlichkeiten zu stark bewerten[26]. Der Brier Score unterscheidet nicht ausreichend zwischen kleinen Änderungen in der Vorhersage. Diese können für seltene Ereignisse von Bedeutung sein. Dementsprechend kann der Brier Score für sehr seltene oder häufige Ereignisse unzureichend sein[3]. Der ECE hat verschiedene Probleme[23][29][1][18]:

- Kann nicht als direktes Optimierungskriterium verwendet werden. Bei der Modelloptimierung wäre die konstant gleichverteilte Vorhersage das Minimum.
- Die Schätzung der Misskalibrierung basiert nur auf dem maximalen Vorhersagewert. Die zweit und dritt höchsten Vorhersagewahrscheinlichkeiten werden allerdings oft mit dem richtigen Klassenlabel assoziiert, dadurch werden viele wahre Vorhersagen nicht bei der Schätzung berücksichtigt.
- Ist ein voreingenommener Schätzer der wahren Kalibrierung (Bias-Varianz-Tradeoff). Die Anzahl der Klassen(Bins) bestimmt wie viele Vorhersagen in jeder Klasse enthalten sind. Je größer die Anzahl, desto größer wird die Varianz innerhalb der Klassen und desto kleiner der Bias des ECE.

Der TACE besitzt einen Bias der bei unterschiedlichen Modellen variiert[29]. In empirischen Versuchen wurde festgestellt, dass der TACE anfällig gegenüber seinen Parametern ist und beim Variieren dieser, keine konstante Rangfolge beim Vergleich unterschiedlicher Modelle garantiert[1].

# Kapitel 6

# Experimente und Ergebnisse

## 6.1 Modell Training

Der Datensatz wird zunächst in eine Trainings- und eine Testmenge aufgeteilt. Der Testdatensatz enthält 25% der verfügbaren Daten, d.h. bei der Vollüberdeckung 4.140.000 und bei der Teilüberdeckung 3.321.200 Beispiele. Der Trainingsdatensatz enthält 75% der Daten, die Vollüberdeckung enthält dann 12.420.000 und die Teilüberdeckung 9.963.600 Beispiele.

Bei der Optimierung wird *stochastischer Gradientenabstieg (SGD)* mit einer Batch-Größe von 256 angewendet. Die Datensätze sind sehr groß, weshalb bereits 10 Epochs beim Training ausreichen. Die initiale Lernrate von 0.1 wird jeweils ab dem 2., 5. und 8. Epoch bei einem Lernratenverlust von $\delta = 10^{-1}$ verringert.

In den Experimenten werden 3 Modelle verglichen: Base, Deep Ensemble und MC-Dropout. Das Base Modell ist ein einzelnes CNN und nutzt die gleiche Architektur wie die Ensemblemitglieder. In den Experimenten soll es als eine Baseline fungieren. Das Ensemble besteht aus 10 Netzwerken und verwendet beim Training Bagging. Das MC-Dropout Modell benutzt eine Dropout-Rate von 0.3 und eine L2 Regularisierung mit dem Gewichtsverlust $\lambda = 10^{-5}$.

Falls die Modelle nachträglich kalibriert werden, wird die im Kapitel 3.6 vorgestellte Technik der Testzeit-Kreuzvalidierung verwendet. Der Temperaturskalar wird durch die Minimierung des NLL über den *Limited-memory BFGS (L-BFGS)* Algorithmus gelernt. Dabei wird eine Lernrate von 0.05 und maximal 50 Iterationen pro Optimierungsschritt verwendet.

## 6.2 In-Domain Experimente

In diesem Kapitel wird das Modellverhalten bezüglich In-Domain Daten untersucht. Die allgemeine Performanz des Klassifikators wird über die Genauigkeit bestimmt. Anschließend werden über verschiedene Metriken die In-Domain Unsicherheit und die Vorhersagequa-

lität bewertet. Die Experimente wurden 4 mal ausgeführt und die folgenden Ergebnisse basieren auf dem Mittelwert der Wiederholungen.

### 6.2.1 Performanz der Klassifikation

In diesem Experiment wird die Performanz der Klassifikation überprüft. Die Qualität wird anhand der Genauigkeit gemessen. Über eine Konfusionsmatrix werden auch die einzelnen Klassen näher betrachtet. Für das Unternehmen Achenbach Buschhütten ist eine Genauigkeit im Bereich von $> 99.8 - 99.9\%$ wünschenswert.
**Hypothese:** Das Deep Ensemble und MC-Dropout Modell sollten mit zunehmender Anzahl an Samples[1] eine bessere Genauigkeit erreichen. Es ist davon auszugehen, dass das Base Modell im Vergleich zum Ensemble eine ähnliche oder schlechtere Performanz erreicht. Das Ensemble ist eine Modellkombination und sollte durch die Berücksichtigung mehrerer unabhängiger Modelle (und ihren Hypothesen) die Fehlerrate stärker reduzieren. Es kann davon ausgegangen werden, dass der MC-Dropout schlechter performt als die anderen Modelle. Der aktive Dropout während der Vorhersage sollte die Genauigkeit negativ beeinflussen.

In der Tabelle 6.1 werden die Genauigkeiten für beide Überdeckungen dargestellt. Bei der Vollüberdeckung performen die Modelle mit einer Genauigkeit von ca. $> 99.94\%$ sehr ähnlich. Auf der Teilüberdeckung ist die Performanz etwas schlechter und die Ergebnisse variieren mehr. Das Base und das Ensemble Modell erreichen eine Genauigkeit von ca. 99.85%. Der MC-Dropout erreicht eine verhältnismäßig schlechtere Genauigkeit von ca. 99.74%. Die Performanz der Modellkombinationen verbessert sich bei einer wachsenden Anzahl an Samples etwas. Dieser Vorgang ist bei der Teilüberdeckung deutlicher.

| | Vollüberdeckung | | | Teilüberdeckung | | |
|---|---|---|---|---|---|---|
| # **Samples** | Base | Deep Ensemble | MC-Dropout | Base | Deep Ensemble | MC-Dropout |
| 1 | 99.939 | 99.940 | 99.911 | 99.843 | 99.824 | 99.541 |
| 2 | | 99.941 | 99.933 | | 99.841 | 99.684 |
| 3 | | 99.941 | 99.934 | | 99.847 | 99.713 |
| 4 | | 99.941 | 99.935 | | 99.850 | 99.720 |
| 5 | | 99.941 | 99.936 | | 99.853 | 99.728 |
| 6 | | 99.941 | 99.936 | | 99.853 | 99.734 |
| 7 | | 99.941 | 99.937 | | 99.853 | 99.735 |
| 8 | | 99.941 | 99.937 | | 99.854 | 99.738 |
| 9 | | 99.941 | 99.937 | | 99.854 | 99.740 |
| 10 | | 99.941 | 99.937 | | 99.855 | 99.741 |

**Tabelle 6.1:** Durchschnittliche Genauigkeit (%) der Modelle über eine variierende Anzahl an Samples.

[1] Samples → (Anzahl) Ensemblemitglieder/MC-Dropout Durchgänge

| | Vollüberdeckung | | | Teilüberdeckung | | |
|---|---|---|---|---|---|---|
| **Metrik** | Base | Deep Ensemble | MC-Dropout | Base | Deep Ensemble | MC-Dropout |
| ECE | 0.001 | 0.004 | 0.004 | 0.015 | 0.013 | 0.110 |
| ECE (TS) | 0.015 | 0.012 | 0.010 | 0.009 | 0.014 | 0.059 |
| TACE | 0.005 | 0.004 | 0.008 | 0.004 | 0.012 | 0.058 |
| TACE (TS) | 0.006 | 0.005 | 0.007 | 0.004 | 0.012 | 0.045 |

**Tabelle 6.2:** Der durchschnittliche ECE(%) und TACE(%) mit und ohne Temperatur Skalierung.

Nun werden die individuellen Fehlerklassen genauer betrachtet. Im Anhang befinden sich Konfusionsmatrizes bzgl. der Vollüberdeckung (Abb. 8.3) und der Teilüberdeckung (Abb. 8.4). Es ist wünschenswert, dass die Genauigkeit klassenübergreifend hoch ist. Bei der Vollüberdeckung ist die Performanz auf allen Klassen sehr gut, wobei der SIKO Fehler verhältnismäßig etwas schlechter abschneidet. Er wurde 2326 mal mit der normalen Klasse verwechselt. Auf der Teilüberdeckung ist die Performanz ebenfalls sehr gut. Die meisten Fehler wurden in der normalen und der Sensorfehler Klasse gemacht. Die Ursache ist wahrscheinlich auf das sich ähnelnde Signal zurückzuführen. Je weniger ein Sensor vom Band überdeckt ist, desto kleiner wird seine Amplitude im Signal. Ein ähnliches Verhalten kann man beim Sensorfehler selbst beobachten, vor allem bei niedrigen Grenzfrequenzen.

### 6.2.2 Modell Kalibrierung

In diesem Experiment wird die Kalibrierung vor und nach einer Temperaturskalierung verglichen. Die Qualität wird mithilfe der beiden Metriken Expected Calibration Error und Thresholded Adaptive Calibration Error approximiert. Aus den Ergebnissen soll gefolgert werden, ob die Modelle kalibrierte Vorhersagen erstellen oder ob eine nachträgliche Kalibrierung der Modelle notwendig ist.
**Hypothese:** Falls die Modelle schlecht kalibriert sind, sollte die Temperaturskalierung den ECE und TACE verringern.

In Abbildung 6.1 und 8.5 wird die Qualität der Kalibrierung über Vertrauenshistogramme und Zuverlässigkeitsdiagramme[6][22] visualisiert. Das Histogramm stellt die Verteilung des vorhergesagten Vertrauens dar. In den Abbildungen kann man sehen, dass sich die Modelle bei den meisten Beispielen sehr sicher sind. Die überwiegende Mehrheit besitzt ein Vertrauen $> 99\%$. Das Zuverlässigkeitsdiagramm visualisiert die Unterschiede zwischen der Genauigkeit und dem Vertrauen innerhalb der Klassen(Bins). Die gestrichelte Linie stellt die erwartete Genauigkeit dar. Der Abstand zwischen der Genauigkeit und dem Vertrauen jeder Klasse sollte möglichst klein sein, dann repräsentiert das Vertrauen die erwartete Genauigkeit. Wie man in den Abbildungen sehen kann, ist die größte Klasse (Vertrauen $> 90\%$) sehr gut kalibriert. Vorhersagen, die ein kleineres Vertrauen besitzen sind etwas schlechter kalibriert. Bei ihnen handelt es sich aber nur um einen sehr kleinen Anteil der Gesamtdaten. Das durchschnittliche Vertrauen und die durchschnittliche Genauigkeit sind

sich sehr ähnlich, zusätzlich ist der ECE und der TACE (Tabelle 6.2) niedrig. Diese Faktoren lassen darauf schließen, dass die Modelle auch ohne nachträgliche Skalierung gut kalibriert sind. Tatsächlich verschlechtert die Temperaturskalierung beim Deep Ensemble und generell bei der Vollüberdeckung die Kalibrierung etwas. Bei der Optimierung des Temperaturskalars wurden verschiedene Lernraten ausprobiert, aber die Ergebnisse waren vergleichbar. Möglicherweise sind zu viele Beispiele mit hohem, berechtigten Vertrauen in den Batches enthalten, sodass der Skalar eher zu ihren Gunsten optimiert wird.

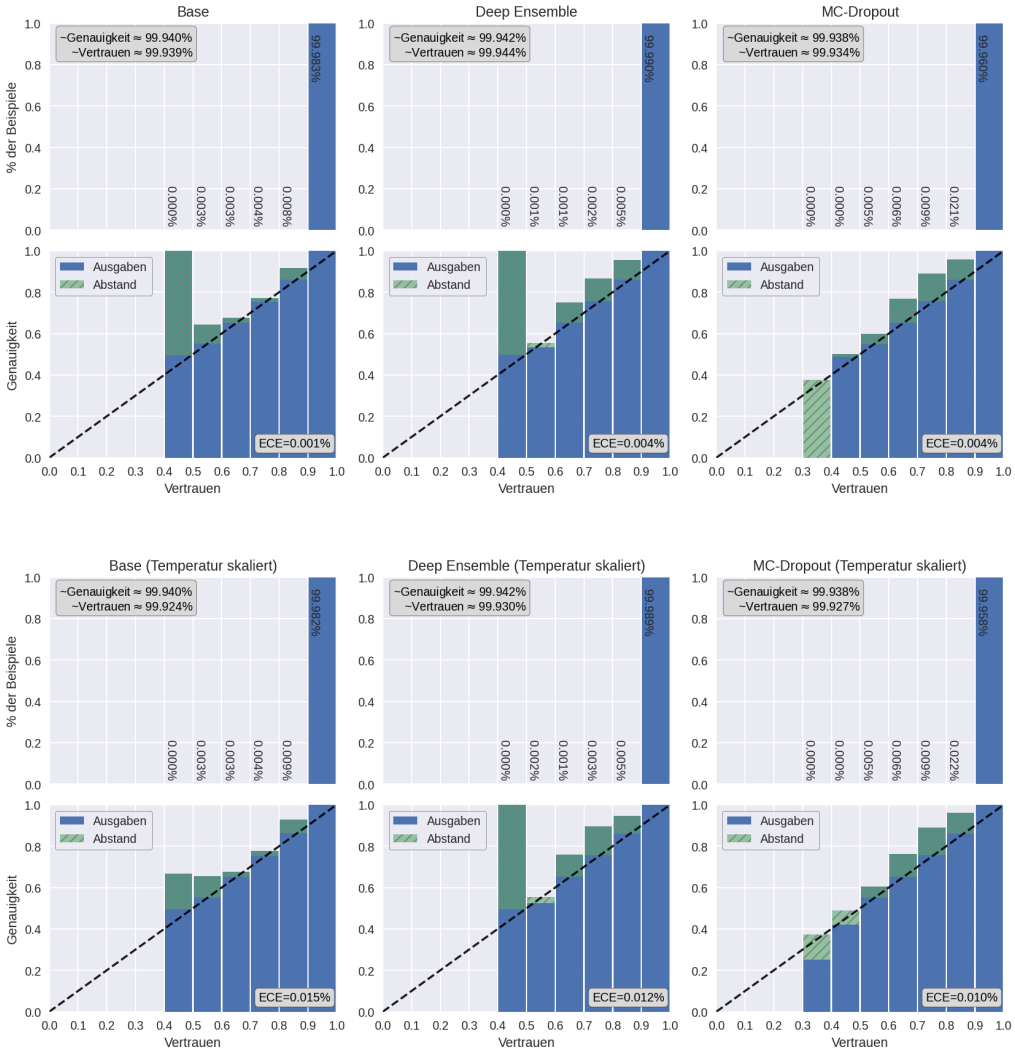


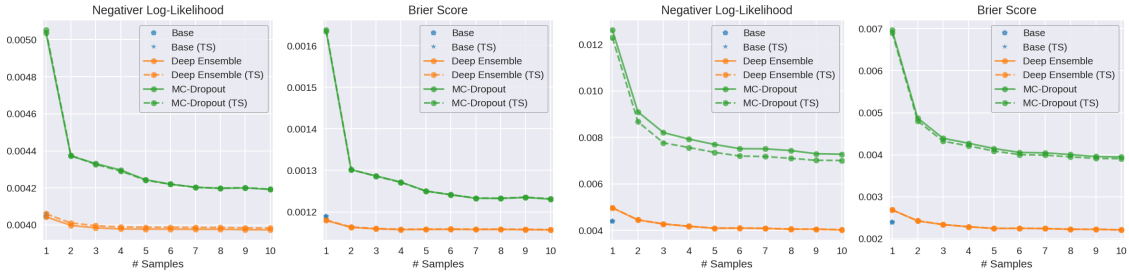

**Abbildung 6.1:** Vollüberdeckung: Vertrauens-Histogramme (oben) und Zuverlässigkeitsdiagramme (unten) des ECE mit und ohne Temperatur Skalierung.

### 6.2.3 In-Domain Unsicherheit

Der NLL und der Brier Score sind beides Metriken zur Schätzung der In-Domain Unsicherheit. In diesem Experiment werden die beiden Metriken über eine wachsende Anzahl

an Samples verglichen.

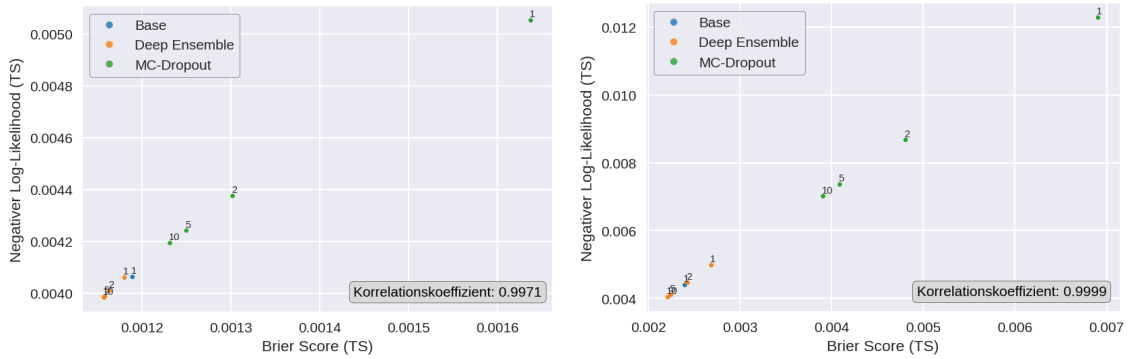
**Hypothese:** Die Modellkombinationen sollten mit einer zunehmender Anzahl an Samples sicherer werden.




**(a)** Vollüberdeckung **(b)** Teilüberdeckung

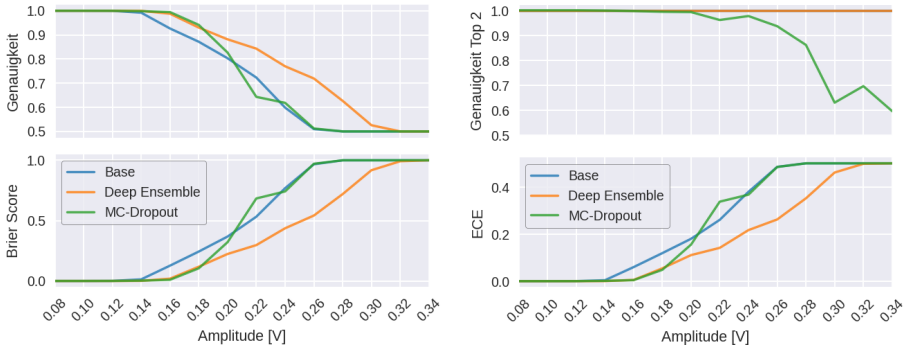
**Abbildung 6.2:** Negativer Log-Likelihood und Brier Score über eine variierende Anzahl an Samples mit und ohne Temperatur Skalierung.

In der Abbildung 6.2 ist der Verlauf der beiden Metriken dargestellt. Die Ergebnisse unterstützen die im vorherigen Kapitel erlangten Erkenntnisse. Der niedrige NLL und Brier Score zeigen, dass beide Modelle den richtigen Vorhersagen hohes Vertrauen zuordnen. Mit einer zunehmenden Anzahl an Samples fallen beide Metriken. Dieser Vorgang ist beim MC-Dropout deutlicher erkennbar. Das Ensemble scheint nur bei den ersten 2-3 Samples an Sicherheit zu gewinnen. In Abbildung 6.3 kann man sehen, dass der NLL und der Brier Score einen starken linearen Zusammenhang haben. Auf der Vollüberdeckung erreichen die Metriken einen Korrelationskoeffizienten von 0.997 und auf der Teilüberdeckung 0.999.




**(a)** Vollüberdeckung **(b)** Teilüberdeckung

**Abbildung 6.3:** Negativer Log-Likelihood vs Brier Score, über eine variierende Anzahl an Samples mit Temperaturskalierung.




**(a)** Genauigkeit vs Brier Score **(b)** Top 2 Genauigkeit vs ECE

**Abbildung 6.4:** Teilüberdeckung: Abgebildet sind die Genauigkeit, der Brier Score, die Top 2 Genauigkeit und der erwartete Kalibrierungsfehler(ECE) bei einer kovariaten Verschiebung der Fehlerklasse „Dirty Gap“.

## 6.3 Out-Of-Domain Experimente

In diesem Kapitel werden Experimente durchgeführt, die das Modellverhalten bei einer Datensatzverschiebung untersuchen. Die Datensatzverschiebung ist ein Problem das auftritt, wenn die multivariate Verteilung der Ein- und/oder Ausgaben zwischen der Trainings- und Testphase unterschiedlich sind. Ein spezieller Fall der Datensatzverschiebung ist die kovariate Verschiebung, hier verändert sich nur die Eingabeverteilung. Im Folgenden wird von einem Out-Of-Domain Fall gesprochen, wenn beide Verteilungen unterschiedlich sind.

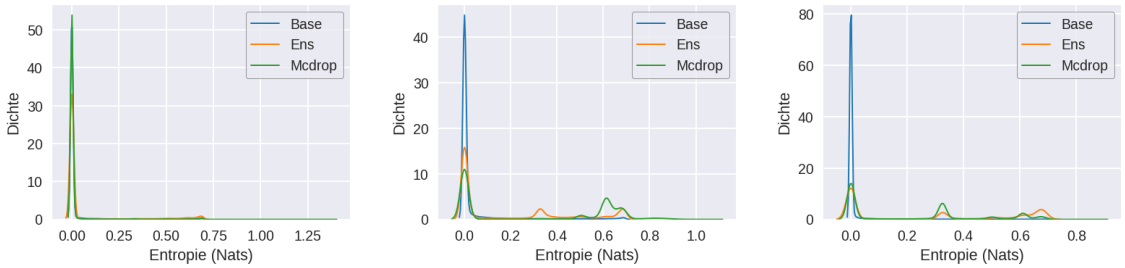
### 6.3.1 Kovariate Verschiebung

In diesem Experiment wird die Reaktion der Modelle auf eine kovariate Verschiebung untersucht. Für das Experiment eignet sich die Fehlerklasse „Dirty Gap“. Hier kann eine Verschiebung über die Erhöhung der Amplitudenstärke simuliert werden. Im Training wurden Stärken zwischen 0.04 bis $0.08V$ berücksichtigt. Nun werden sie von 0.10 auf $0.34V$ erhöht. Auf der untersuchten Verschiebung haben die Modelle der Vollüberdeckung unverändert die richtigen Vorhersagen mit hohen Vertrauen getätigt. Deshalb werden im Folgenden nur die Resultate[2] der Teilüberdeckung betrachtet.

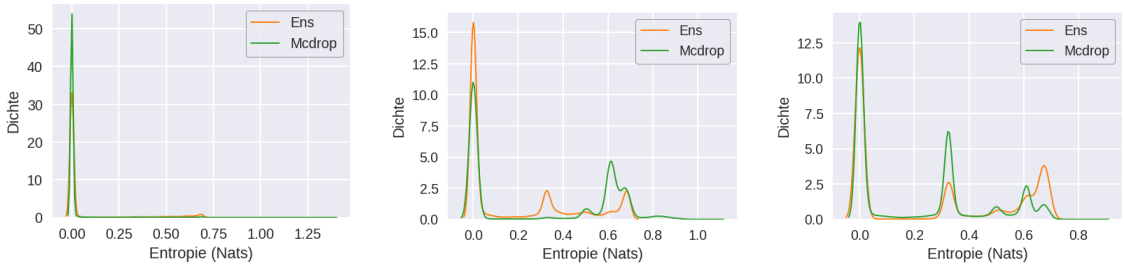
**Hypothese:** Je größer die Verschiebung ist, desto schlechter sollte die Genauigkeit und das mit den Vorhersagen assoziierte Vertrauen werden. D.h. es ist wünschenswert, dass die (gute) Kalibrierung (siehe Kapitel 6.2.2) möglichst lange auf der wachsenden Verschiebung hält. Falls die Vorhersagequalität abnimmt, sollten die Modelle über eine erhöhte Vorhersageentropie ihre Unsicherheit beachten.

Wie man in Abbildung 6.4a sehen kann, bleibt die Genauigkeit auf allen Modellen zunächst stabil. Erst bei einer Amplitudenstärke von $0.14 - 0.16V$ verschlechtern sich die

[2]Die vorgestellten Ergebnisse basieren auf Eingaben der Fehlerklasse „Dirty Gap“.




**(a)** PE bei $DG_{Amplitude} = 0.16V$ **(b)** PE bei $DG_{Amplitude} = 0.22V$) **(c)** PE bei $DG_{Amplitude} = 0.26V$




**(d)** MI bei $DG_{Amplitude} = 0.16V$ **(e)** MI bei $DG_{Amplitude} = 0.22V$ **(f)** MI bei $DG_{Amplitude} = 0.26V$

**Abbildung 6.5:** Teilüberdeckung: Kerndichteschätzung der Vorhersageentropie(PE) und der gegenseitige Information(MI) bei unterschiedlich starken Amplitudenstärken $DG_{Amplitude}$.

Vorhersagen. Also im Vergleich zur Trainingskonfiguration, eine ungefähr doppelt so große Schwingungsamplitude. Das Ensemble erreicht bei größeren Verschiebungen eine bessere Genauigkeit als die anderen Modelle. Wenn man die 2 wahrscheinlichsten Vorhersagen berücksichtigt, performt der MC-Dropout zunehmend schlechter. Hingegen erreichen das Base und Ensemble Modell eine 100% Top 2 Genauigkeit bei allen Verschiebungen. Der Brier Score und der erwartete Kalibrierungsfehler sehen sich sehr ähnlich. Ihre Verläufe lassen sich gut mit dem der Genauigkeit abgleichen. Auch hier gilt, ab einer Amplitudenstärke von ca. $0.16V$ verschlechtert sich die Vorhersagequalität. Mit einer zunehmend schlechter werdenden Genauigkeit steigt der Kalibrierungsfehler, bzw. die Vorhersagen besitzen ein unangemessenes Vertrauen. Interessant ist, dass die Metriken sich ab einer bestimmten Verschiebung nicht mehr verändern. Dieser Punkt wird beim Base und MC-Dropout Modell eher erreicht als beim Ensemble. Um den Verlauf besser zu verstehen, werden die Verschiebungen der Stärken $0.16, 0.22$ und $0.26V$ näher untersucht.

In Abbildung 6.5 ist die Vorhersageentropie und die gegenseitige Information der Modelle über einen Kerndichteschätzer visualisiert[3]. Das Base Modell ist nicht in der Lage seine eigene Unsicherheit zu berücksichtigen, bei allen Verschiebungen ist die Vorhersageentropie sehr niedrig. Beim Ensemble und MC-Dropout steigt die Vorhersageentopie bei größe-

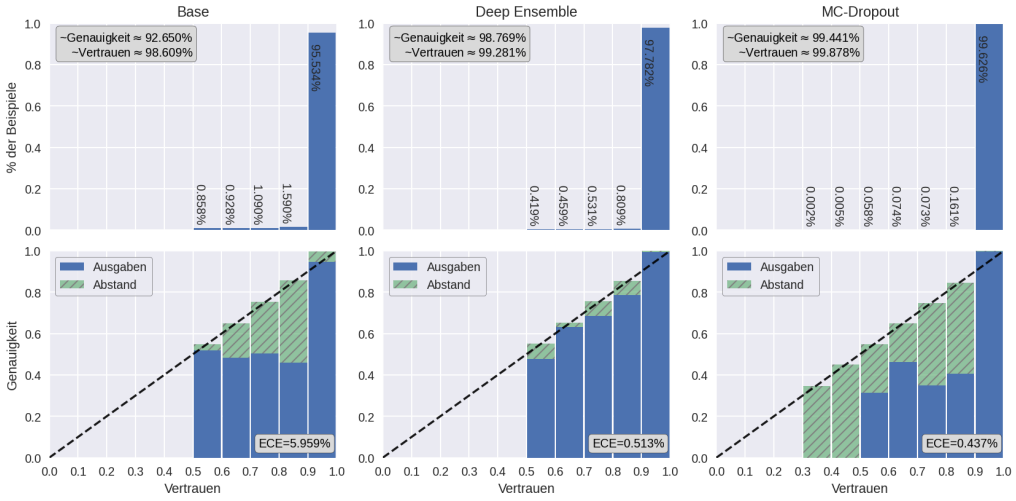
[3]Die gegenseitige Information wird u.a. über die erwartete Vorhersageentropie approximiert. Bei $\#Samples < 2$ oder für Modellausgaben die nicht variieren(Base Modell), besitzen die Ergebnisse eine zu geringe Varianz und lassen sich nicht über eine KDE visualisieren.

ren Verschiebungen an. Sie ist aber mit $0.6 - 0.8$ Nats immer noch relativ niedrig. Der MC-Dropout erreicht zunächst eine höhere Entropie als das Ensemble (Abb. 6.5b, 6.5e), möglicherweise weil seine Vorhersagequalität sich schneller verschlechtert.
In Abbildung 6.6 wird die Kalibrierung visualisiert. Man kann erkennen, dass das durchschnittliche Vertrauen bei allen Modellen zunächst leicht abnimmt. Insgesamt wird aber selbst bei falschen Vorhersagen ein hohes Vertrauen ausgegeben. Das Deep Ensemble bleibt im Vergleich am besten kalibriert. Bei starken Verschiebungen, bspw. $DG_{Amplitude} = 0.32V$, wird das gesamte Vertrauen zur Hälfte mit der richtigen oder einer falsche Klassen assoziiert (Abb. 8.6).

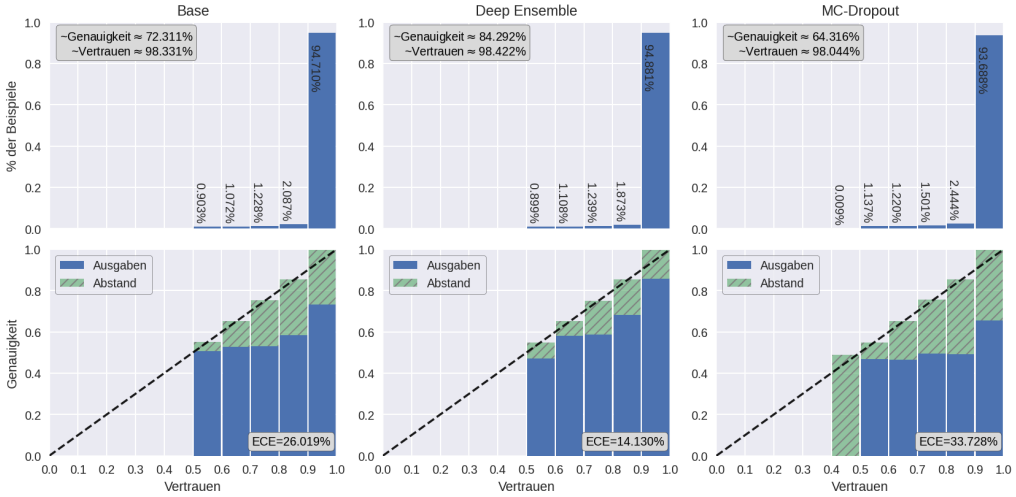
### 6.3.2 Vollständig Out-Of-Domain

In diesem Experiment wird die Reaktion der Modelle auf Out-Of-Domain Daten getestet. Die OOD Daten werden über die bekannten Klassen simuliert. Dazu wird beim Training jeweils eine der Klassen aus dem Beispieldatensatz entfernt und dementsprechend nicht in der Trainingsphase verwendet. In der Testphase wird auf den Beispielen der OOD Klasse die Entropie berechnet.
**Hypothese:** Falls das Modell „merkt“, dass es mit unbekannten Daten konfrontiert wird, sollte es in Form einer hohen Entropie seine Unsicherheit berücksichtigen.

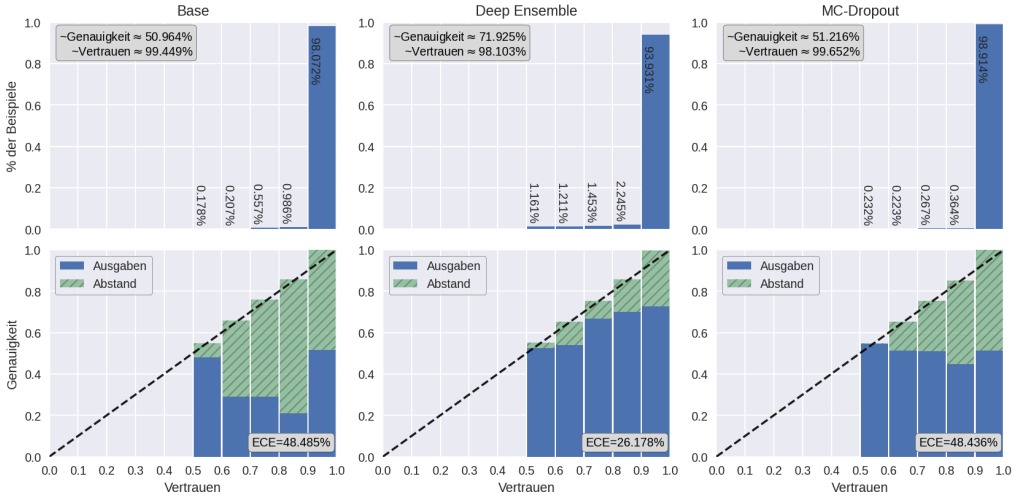
In Abbildung 6.7 wird die Vorhersageentropie und in Abbildung 6.8 die gegenseitige Information der unterschiedlichen OOD Klassen visualisiert. Mann kann erkennen, dass bei allen Klassen die Modelle eine überwiegend geringe Entropie aufweisen, wobei das Ensemble und der MC-Dropout etwas unsicherer sind. In den Klassen „Normal“ und „Rauschen“ der Vollüberdeckung ist die Entropie leicht erhöht. Insgesamt sind aber alle Modelle fälschlicherweise zuversichtlich.




**(a)** $DG_{Amplitude} = 0.16V$

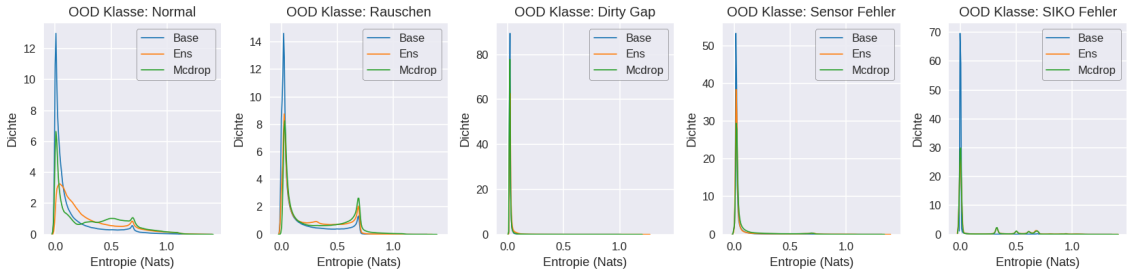



**(b)** $DG_{Amplitude} = 0.22V$

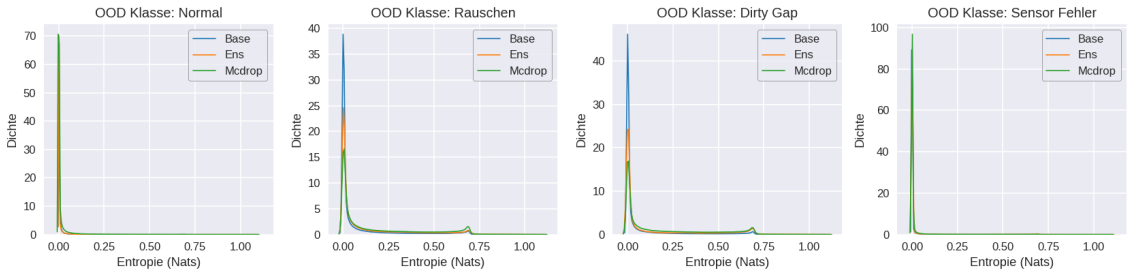



**(c)** $DG_{Amplitude} = 0.26V$

**Abbildung 6.6:** Teilüberdeckung: Vertrauens-Histogramme (oben) und Zuverlässigkeitsdiagramme (unten) des ECE bei unterschiedlich starken Datensatzverschiebung der Klasse „Dirty Gap“.

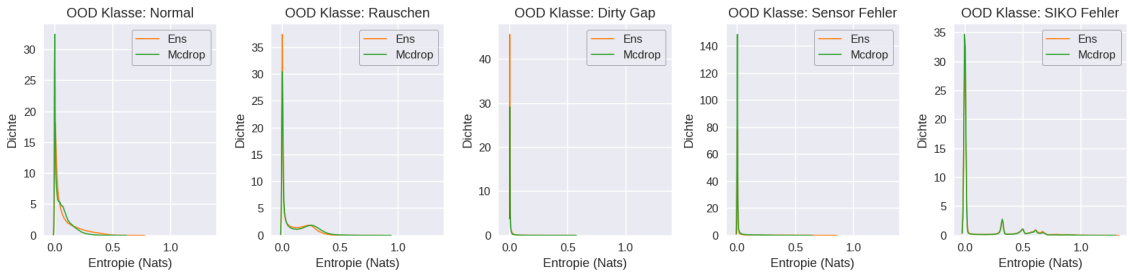



**(a)** Vollüberdeckung

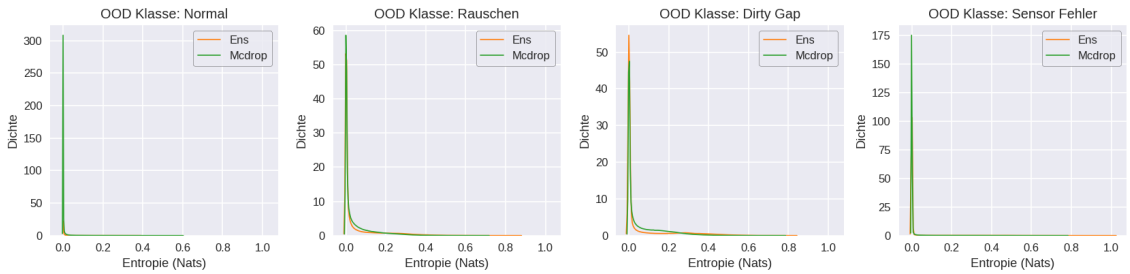



**(b)** Teilüberdeckung

**Abbildung 6.7:** Kerndichteschätzung der Vorhersageentropie bei unterschiedlichen OOD Klassen.




**(a)** Vollüberdeckung




**(b)** Teilüberdeckung

**Abbildung 6.8:** Kerndichteschätzung der gegenseitigen Information bei unterschiedlichen OOD Klassen.

# Kapitel 7

# Fazit

In dieser Arbeit wurde erfolgreich ein Neuronales Netzwerk entwickelt, dass auf den simulierten Daten einer Planheitsmessrolle die Fehlerfälle weißes Rauschen, Dirty Gap, Sensorfehler und SIKO-Fehler klassifizieren kann. Das Modell wurde jeweils durch die Methoden Deep Ensemble und MC-Dropout erweitert. Ziel war dabei zu untersuchen, ob die Verfahren eine bessere Unsicherheitsschätzung bei In- und Out-Of-Domain Daten ermöglichen.

Am Anfang der Arbeit wurden allgemeine Grundlagen gegeben. Dazu gehört u.a. eine Beschreibung des Walzvorganges, damit der Leser das betrachtete Anwendungsproblem besser nachvollziehen kann. Als nächstes wurde in den technischen Grundlagen die Problemstellung definiert, Informationen zur Unsicherheitsbestimmung gegeben und die verschiedenen Komponenten und Verfahren der Modelle eingeführt. Zur Datenerhebung wurde der Messrollen Signal Simulator verwendet. Der Simulator wurde mit zusätzlichen Funktionalitäten erweitert. Diese ermöglichen einen einfacheren und effektiveren Datenerhebungsprozess. Im letzten Kapitel wurden verschiedene Experimente durchgeführt. Es wurde die allgemeine Performanz der Klassifikation und die Modellunsicherheit bei In- und Out-Of-Domain Daten getestet.

Die Performanz der Klassifikation ist sehr gut, die Modelle erreichen auf allen Klassen mit ca. $99.8 - 99.9\%$ eine hohe Genauigkeit. Auf Basis der In-Domain Daten erstellen die Modelle kalibrierte Vorhersagen, eine nachträgliche Skalierung ist nicht notwendig. Der NLL und Brier Score lassen darauf schließen, dass die In-Domain Unsicherheit niedrig ist. Insgesamt haben das Base und Ensemble Modell in den In-Domain Experimenten etwas besser abgeschnitten.
Eine kovariate Verschiebung wurde über die Fehlerklasse „Dirty Gap“ simuliert. Während bei den anfänglichen Verschiebungen alle Modelle erfolgreich generalisieren konnten, performte das Ensemble bei stärkeren Verschiebungen am besten. Es blieb im Vergleich besser kalibriert und erreichte höhere Genauigkeiten. Der MC-Dropout und das Deep Ensemble konnten über eine erhöhte Vorhersageentropie ihre Unsicherheit besser berücksichtigen

als das Base Modell. Bei den komplett OOD Daten waren alle Modelle fälschlicherweise zuversichtlich.

Zusammenfassend kann festgehalten werden, dass die Modelle sich für das reine Klassifikationsproblem eignen. Das Deep Ensemble hat im Vergleich die besten Ergebnisse erreicht. Ob die Vorhersageentropie in der realen Anwendung verwendet werden kann, muss weiter untersucht werden. Vielleicht können schon kleinere Entropieausgaben als ausreichende Unsicherheit gewertet werden.

# Kapitel 8

# Weitere Informationen

## 8.1 Abbildungen

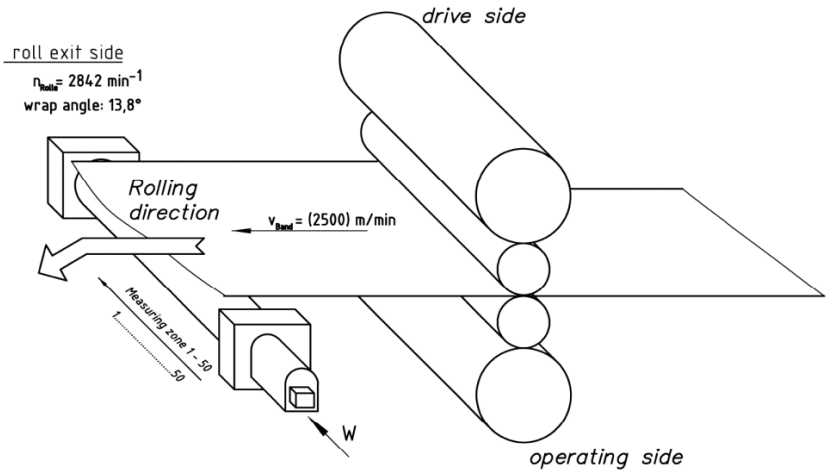



**Abbildung 8.1:** Visualisierung des Walz- und Messprozesses.

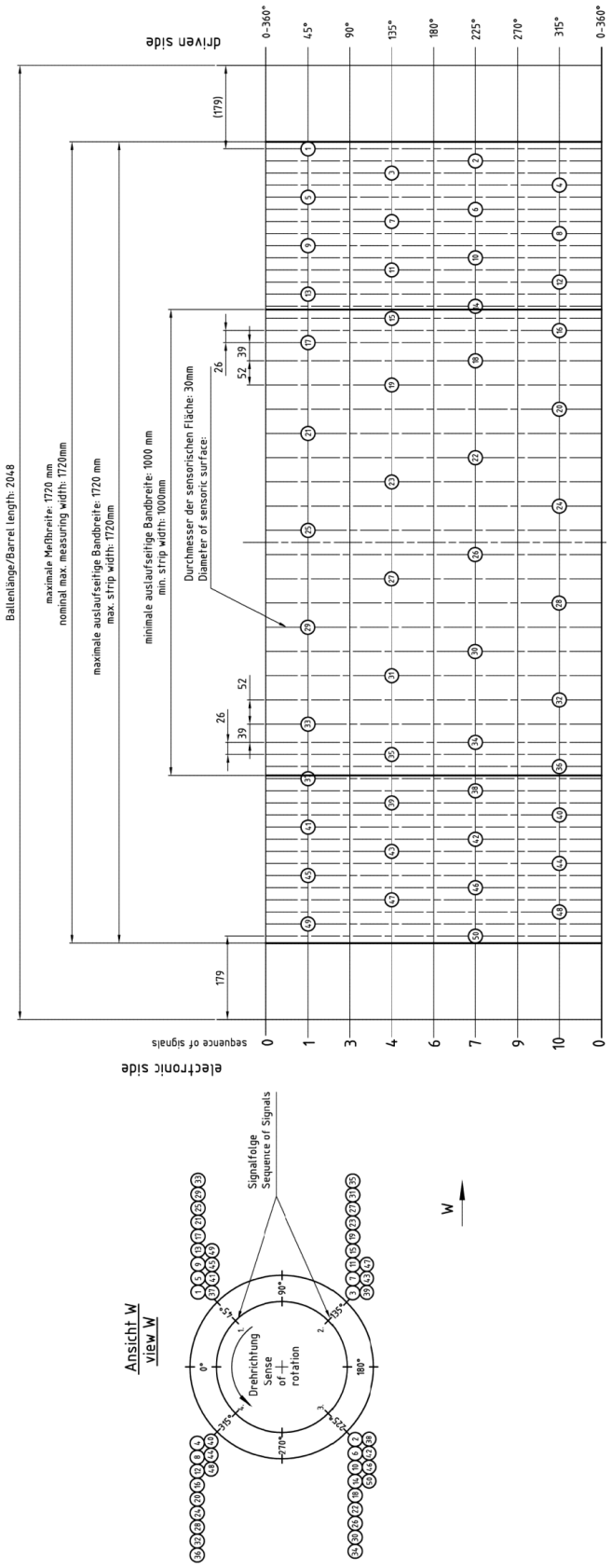
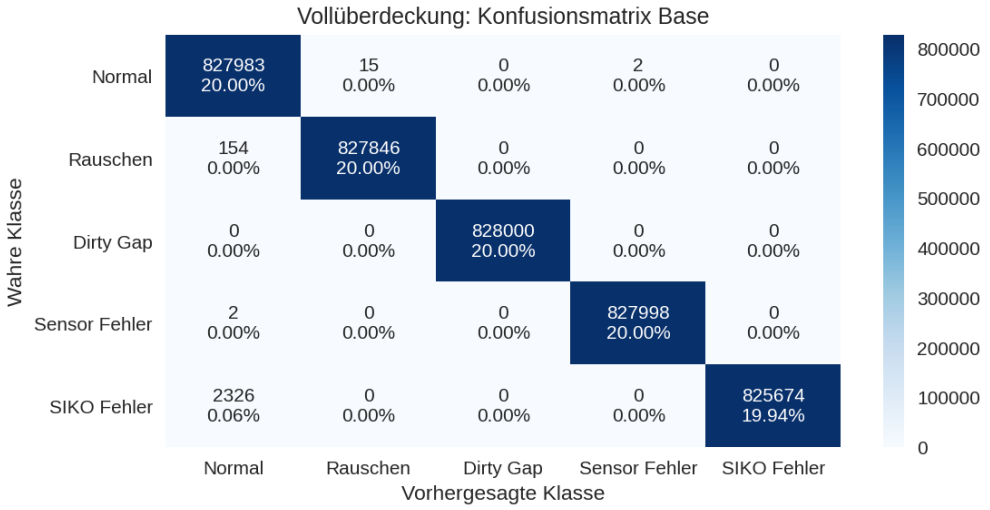
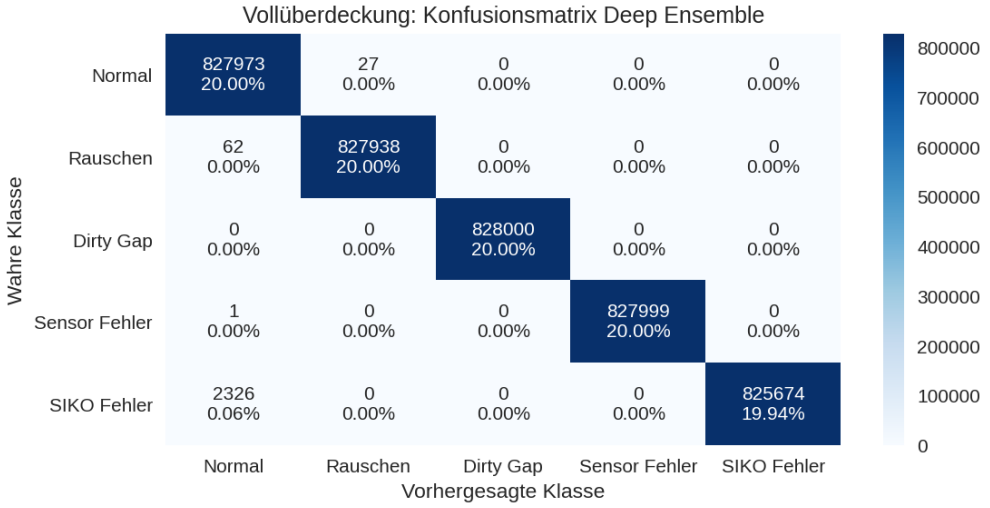


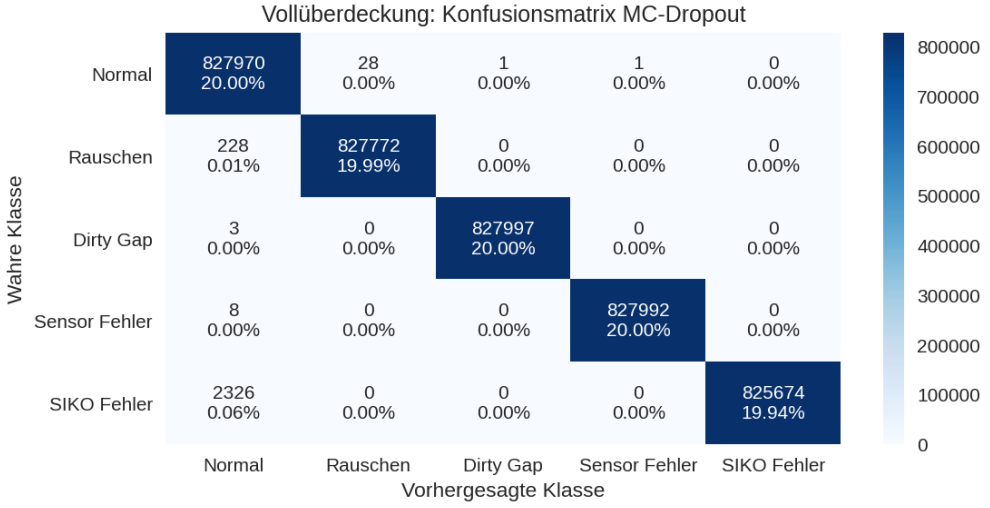

**Abbildung 8.2:** Schema der Planheitsmessrolle.




**(a)** Base

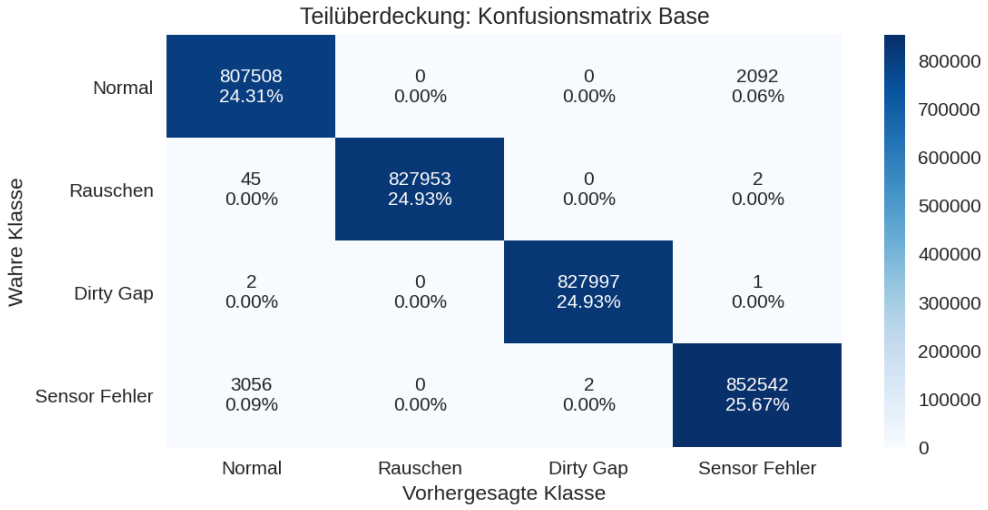



**(b)** Deep Ensemble

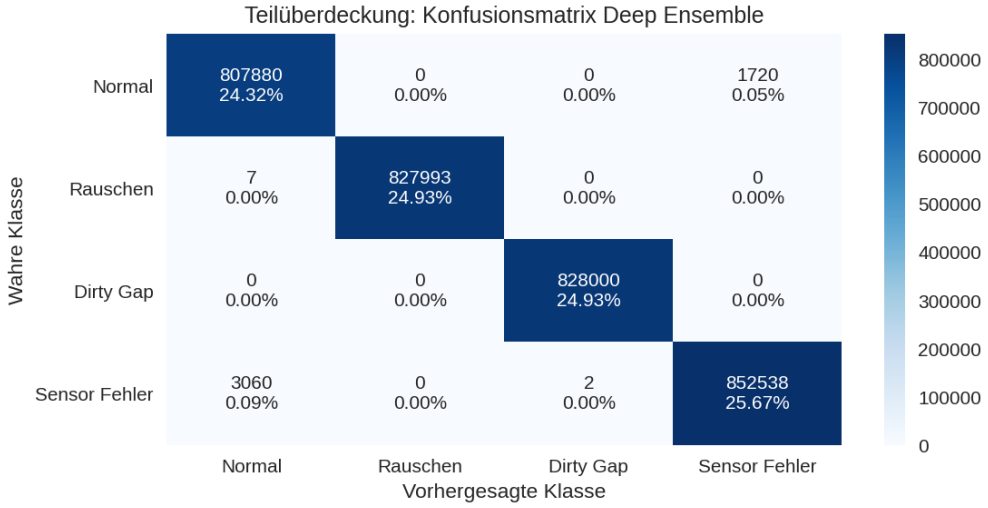



**(c)** MC-Dropout

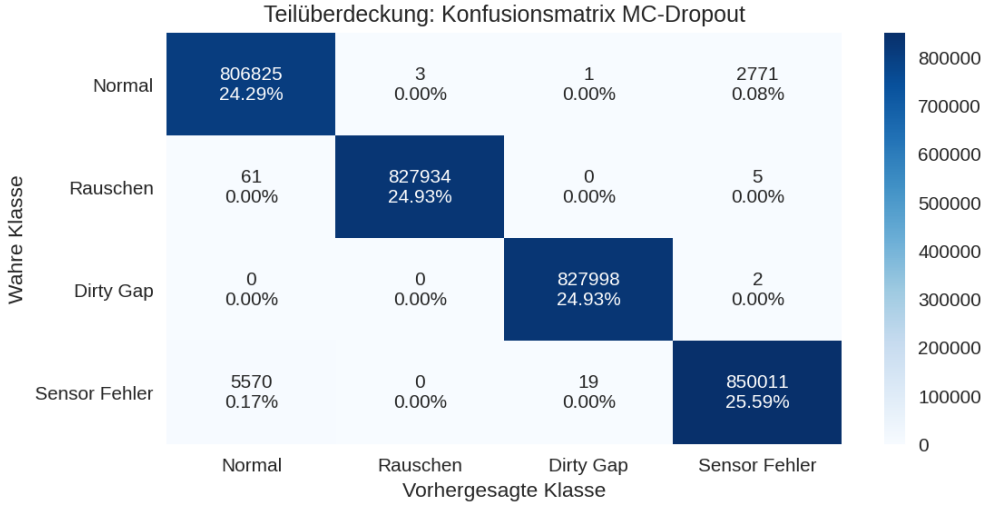
**Abbildung 8.3:** Vollüberdeckung: Konfusionsmatrix der Klassifikation.




**(a)** Base




**(b)** Deep Ensemble




**(c)** MC-Dropout

**Abbildung 8.4:** Teilüberdeckung: Konfusionsmatrix der Klassifikation.

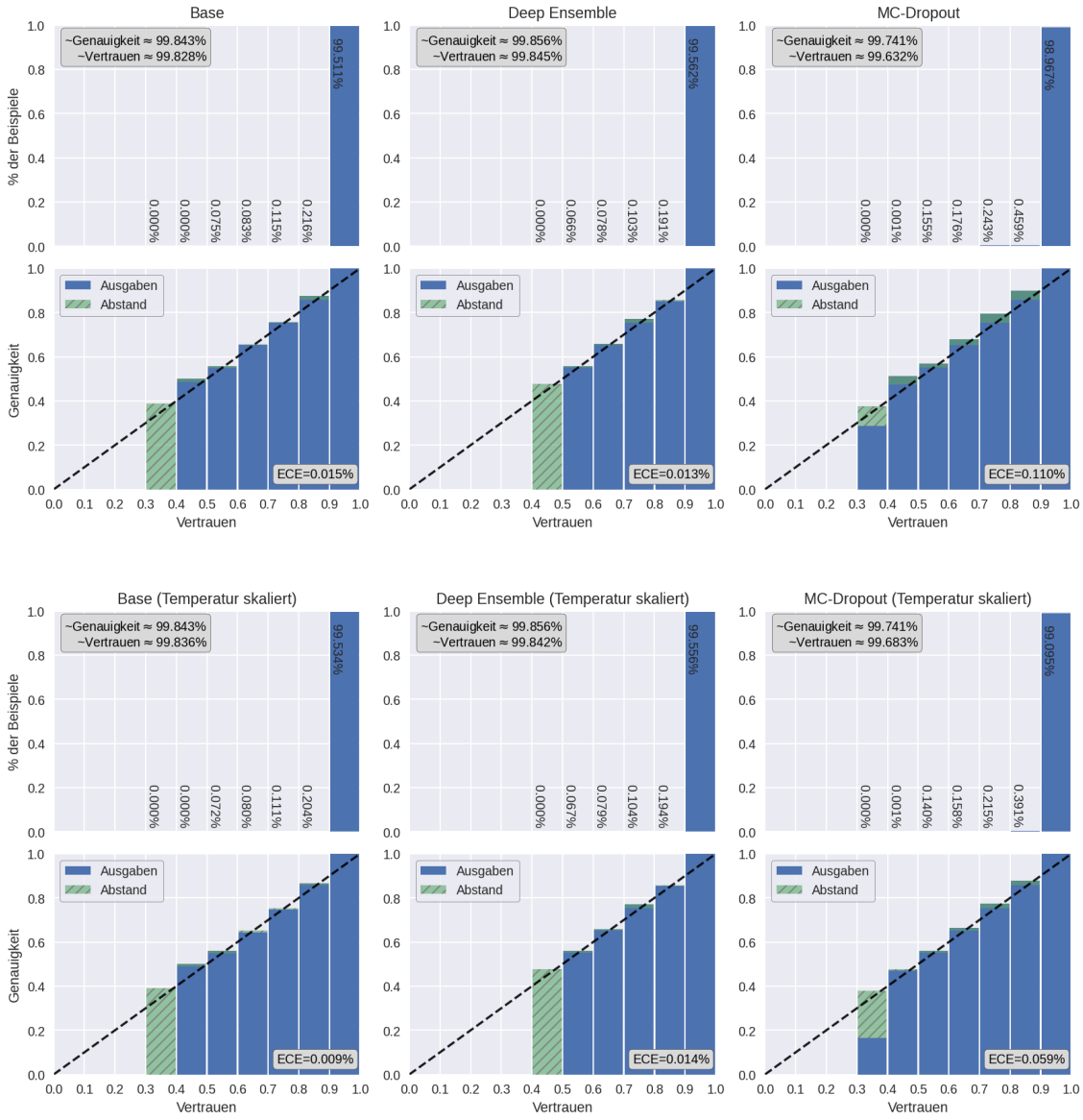



**Abbildung 8.5:** Teilüberdeckung: Vertrauenshistogramme (oben) und Zuverlässigkeitsdiagramme (unten) des ECE mit und ohne Temperatur Skalierung.

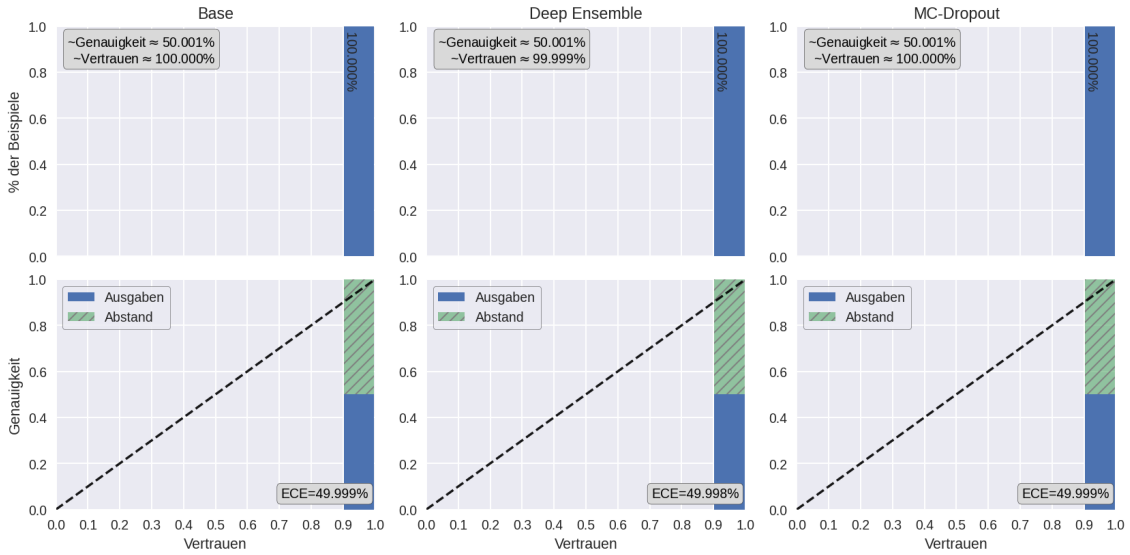



**Abbildung 8.6:** Teilüberdeckung: Vertrauenshistogramme (oben) und Zuverlässigkeitsdiagramme (unten) bei einer Datensatzverschiebung der Klasse „Dirty Gap“. Abgebildet sind die Ergebnisse bei einer Amplitudenstärke von $DG_{Amplitude} = 0.34V$.

# Abbildungsverzeichnis

# Literaturverzeichnis


[1] Ashukha, Arsenii, Alexander Lyzhov, Dmitry Molchanov und Dmitry Vetrov: *Pitfalls of In-Domain Uncertainty Estimation and Ensembling in Deep Learning*, 2020.

[2] Aslam, Javed A., Raluca A. Popa und Ronald L. Rivest: *On Estimating the Size and Confidence of a Statistical Audit.* In: *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, EVT'07, Seite 8, USA, 2007. USENIX Association.

[3] Benedetti, Riccardo: *Scoring Rules for Forecast Verification.* Monthly Weather Review, 138(1):203 – 211, 01 Jan. 2010.

[4] Brando, Axel, José A. Rodríguez-Serrano, Mauricio Ciprian, Roberto Maestre und Jordi Vitrià: *Uncertainty Modelling in Deep Networks: Forecasting Short and Noisy Series.* CoRR, abs/1807.09011, 2018.

[5] Brier, Glenn W.: *Verification of Forecasts Expressed in Terms of Probability.* Monthly Weather Review, 78(1):1, Januar 1950.

[6] DeGroot, Morris H. und Stephen E. Fienberg: *The Comparison and Evaluation of Forecasters.* Journal of the Royal Statistical Society. Series D (The Statistician), 32(1/2):12–22, 1983.

[7] Dennis Preljević, Roger Feist, Rannam Chaaban: *MRSS - Messrollen Signal Simulator*, 2020.

[8] Dietterich, Thomas G.: *Ensemble Methods in Machine Learning.* In: *Multiple Classifier Systems*, Seiten 1–15, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[9] Fawaz, Hassan Ismail, Germain Forestier, Jonathan Weber, Lhassane Idoumghar und Pierre-Alain Muller: *Deep learning for time series classification: a review.* CoRR, abs/1809.04356, 2018.

[10] Gal: *MC-Dropout Model unceirtanty.* `http://www.cs.ox.ac.uk/people/yarin.gal/website/blog_2248.html` Besucht: 2020-09-09.

[11] Gal, Yarin: *Uncertainty in Deep Learning*. Doktorarbeit, University of Cambridge, 2016.

[12] Goodfellow, Ian, Yoshua Bengio und Aaron Courville: *Deep Learning*. The MIT Press, 2016.

[13] Gundersen, Kristian, Guttorm Alendal, Anna Oleynik und Nello Blaser: *Binary Time Series Classification with Bayesian Convolutional Neural Networks When Monitoring for Marine Gas Discharges*. Algorithms, 13:145, 06 2020.

[14] Guo, Chuan, Geoff Pleiss, Yu Sun und Kilian Q. Weinberger: *On Calibration of Modern Neural Networks*, 2017.

[15] Houlsby, Neil, Ferenc Huszár, Zoubin Ghahramani und Máté Lengyel: *Bayesian Active Learning for Classification and Preference Learning*, 2011.

[16] Ioffe, Sergey und Christian Szegedy: *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. In: Bach, Francis und David Blei (Herausgeber): *Proceedings of the 32nd International Conference on Machine Learning*, Band 37 der Reihe *Proceedings of Machine Learning Research*, Seiten 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[17] Kana, Michel: *Aleatoric and epistemic uncertainty*. `https://miro.medium.com/max/788/1*5vj9r-scd3fEKHRXnqqurg.png` Besucht: 2021-04-09.

[18] Kumar, Ananya, Percy S Liang und Tengyu Ma: *Verified Uncertainty Calibration*. In: Wallach, H., H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox und R. Garnett (Herausgeber): *Advances in Neural Information Processing Systems*, Band 32. Curran Associates, Inc., 2019.

[19] Lakshminarayanan, Balaji, Alexander Pritzel und Charles Blundell: *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles*. In: Guyon, I., U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan und R. Garnett (Herausgeber): *Advances in Neural Information Processing Systems 30*, Seiten 6402–6413. Curran Associates, Inc., 2017.

[20] Murphy, Allan H.: *A New Vector Partition of the Probability Score*. Journal of Applied Meteorology and Climatology, 12(4):595 – 600, 01 Jun. 1973.

[21] Naeini, Mahdi Pakdaman, Gregory F Cooper und Milos Hauskrecht: *Obtaining Well Calibrated Probabilities Using Bayesian Binning*. In: *AAAI*, Seite 2901–2907, 2015.

[22] Niculescu-Mizil, Alexandru und Rich Caruana: *Predicting Good Probabilities with Supervised Learning.* In: *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, Seite 625–632, New York, NY, USA, 2005. Association for Computing Machinery.

[23] Nixon, Jeremy, Mike Dusenberry, Ghassen Jerfel, Timothy Nguyen, Jeremiah Liu, Linchuan Zhang und Dustin Tran: *Measuring Calibration in Deep Learning*, 2020.

[24] Pin, G., V. Francesconi, F.A. Cuzzola und T. Parisini: *Adaptive task-space metal strip-flatness control in cold multi-roll mill stands.* Journal of Process Control, 23(2):108–119, 2013. IFAC World Congress Special Issue.

[25] Platt, John C.: *Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods.* In: *ADVANCES IN LARGE MARGIN CLASSIFIERS*, Seiten 61–74. MIT Press, 1999.

[26] Quinonero Candela, J., CE. Rasmussen, F. Sinz, O. Bousquet und B. Schölkopf: *Evaluating Predictive Uncertainty Challenge.* In: *Machine Learning Challenges: Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, Seiten 1–27, Berlin, Germany, April 2006. Max-Planck-Gesellschaft, Springer.

[27] Shannon, C. E.: *A mathematical theory of communication.* The Bell System Technical Journal, 27(3):379–423, 1948.

[28] Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever und Ruslan Salakhutdinov: *Dropout: A Simple Way to Prevent Neural Networks from Overfitting.* Journal of Machine Learning Research, 15(56):1929–1958, 2014.

[29] Vaicenavicius, Juozas, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll und Thomas B. Schön: *Evaluating model calibration in classification*, 2019.

[30] Zhao, Bendong, Huanzhang Lu, Shangfeng Chen, Junliang Liu und Dongya Wu: *Convolutional neural networks for time series classification.* Journal of Systems Engineering and Electronics, 28(1):162–169, 2017.