OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

BACHELORARBEIT

Nicolas Zander

Evaluierung der Fréchet-Radar-Distanz zur Bewertung synthetischer Radar-Daten

5. März 2026

Fakultät:	Informatik und Mathematik
Studiengang:	Bachelor Künstliche Intelligenz und Data Science
Abgabefrist:	10. Februar 2026
Betreuung:	Prof. Dr. Timo Baumann
Zweitbegutachtung:	Prof. Dr. Brijnesh Jain
Externe Betreuung:	Dr. Jörg Reichardt (AUMOVIO Germany GmbH)

Kurzfassung

Die Bewertung synthetischer Radar-Daten stellt eine zentrale Herausforderung im Bereich generativer Modelle dar, da etablierte Qualitätsmetriken wie die Fréchet-Inception-Distance (FID) auf domänenspezifische Merkmalsextraktoren angewiesen sind und nicht ohne Weiteres auf Radar-Daten übertragbar sind. In dieser Arbeit wird die Fréchet-Radar-Distance (FRD) vorgestellt und systematisch evaluiert. Die FRD ist ein an die FID angelehntes Distanzmaß, das zufällige Projektionen als Feature-Extraktor verwendet und dadurch nicht auf domänenspezifisch trainierte Modelle angewiesen ist. Zur Validierung der FRD wird eine kontrollierte Testumgebung mit parametrisierten Pseudo-Radar-Generatoren eingesetzt, die eine gezielte Variation statistischer Eigenschaften der erzeugten Radar-Punktwolken ermöglicht. Als Referenzmaß dient eine approximierte Log-Likelihood, welche die tatsächliche Ähnlichkeit der zugrundeliegenden Verteilungen quantifiziert.

Die experimentellen Ergebnisse zeigen, dass die FRD in diesen kontrollierten Szenarien konsistente, reproduzierbare und erwartungskonforme Bewertungen liefert und mit der approximierten Log-Likelihood korreliert. Damit eignet sich die FRD zur robusten Erfassung struktureller Ähnlichkeiten zwischen Radar-Datenverteilungen.

Aufbauend auf dieser Validierung wird die FRD zur Evaluation U-Net-basierter Diffusionsmodelle für die Generierung synthetischer Radar-Daten eingesetzt. Radar-Daten zeichnen sich durch eine dünnbesetzte, verrauschte und physikalisch geprägte Struktur aus, die sich grundlegend von Bilddaten unterscheidet. Die Ergebnisse zeigen, dass Diffusionsmodelle unter idealisierten Trainingsbedingungen in der Lage sind, diese Eigenschaften zu reproduzieren. Zusätzlich wurde der Einfluss der Modellkapazität analysiert. Dabei lässt sich zeigen, dass ab einer bestimmten Modellgröße kein signifikanter Qualitätsgewinn mehr beobachtbar ist.

Inhaltsverzeichnis

1. Einleitung

1.1. Motivation und Kontext

Autonome Straßenfahrzeuge sind Verkehrssysteme, die ohne menschlichen Eingriff am Straßenverkehr teilnehmen können. In den vergangenen Jahren haben sie an gesellschaftlicher und wissenschaftlicher Relevanz gewonnen [1]. Ein wesentlicher Treiber dieser Entwicklung sind Fortschritte moderner Fahrerassistenzsysteme. Sie übernehmen bereits heute zentrale Aufgaben der Fahrzeugführung. Dazu zählen die Umfeldwahrnehmung, Geschwindigkeits- und Abstandskontrolle sowie die Lenkungs- und Spurführung des Fahrzeugs [2].

Ein zentrales Ziel der Entwicklung autonomer Fahrzeuge ist die Erhöhung der Verkehrssicherheit. Laut dem Global Status Report on Road Safety der Weltgesundheitsorganisation sind Verkehrsunfälle weltweit die häufigste Todesursache für Menschen zwischen 5 und 29 Jahren [3]. Eine Analyse von Winkle [4] zeigt, dass 93,5 % aller Verkehrsunfälle auf menschliches Fehlverhalten zurückzuführen sind. Damit stellt menschliches Fehlverhalten den Hauptfaktor für das Unfallgeschehen dar. In vollständig autonomen Fahrzeugen beeinflusst der Mensch das Fahrverhalten nicht. Somit könnten Unfälle, die durch menschliche Fehler entstehen, reduziert werden. Gleichzeitig entstehen neue Herausforderungen. Diese betreffen insbesondere die technische Zuverlässigkeit und die Fehlertoleranz zunehmend komplexer Fahrzeugsysteme [5].

Damit autonome Fahrzeuge in Zukunft sicher am Straßenverkehr teilnehmen können, müssen sie ihre Umgebung kontinuierlich erfassen und interpretieren. Dies erfolgt durch den Einsatz verschiedener Sensortechnologien, darunter visuelle Sensoren wie Kameras und Light Detection and Ranging (LiDAR) oder auch nicht-visuelle Sensoren wie Radio Detection and Ranging (Radar). Diese Sensoren ermöglichen es dem Fahrzeug, physikalische Eigenschaften der Umwelt, wie Entfernungen, Bewegungen und Objektformen, in elektronische Signale zu überführen [6]. Die gewonnenen Daten werden anschließend mithilfe von Algorithmen verarbeitet, um eine semantische Interpretation der Umgebung zu ermöglichen und situationsabhängige Entscheidungen zu treffen [7].

Fahrerassistenzsysteme nutzen zur Entscheidungsfindung in der Regel Verfahren der Künstlichen Intelligenz (KI). Ertel definiert das Ziel von KI als die Fähigkeit, intelligente Entscheidungen zu treffen [8]. Ziel ist es, eine mit menschlicher Leistung vergleichbare Leistungsfähigkeit zu erreichen oder diese in spezifischen Anwendungsfeldern zu übertreffen. Für einen Großteil der im autonomen Fahren

eingesetzten Algorithmen bildet Maschinelles Lernen (ML) die Grundlage. ML ist ein Teilbereich der KI und umfasst Algorithmen, die Entscheidungen nicht durch explizite Programmierung, sondern durch Analyse und Generalisierung aus Trainingsdaten treffen [9]. Die Leistungsfähigkeit dieser Algorithmen hängt nicht nur von ihrer Architektur und Parametrisierung ab, sondern vor allem von der Qualität, Vielfalt und Repräsentativität der Trainingsdaten. Ein typisches Problem ist die eingeschränkte Reaktionsfähigkeit von ML-Algorithmen auf Situationen, die im Trainingsdatensatz nicht oder nur unzureichend vertreten sind [10]. In Bezug auf autonome Straßenfahrzeuge betrifft dies vor allem seltene oder komplexe Szenarien, etwa extreme Wetterbedingungen wie Regen oder Schnee [11, 12] oder ungewöhnliche Verkehrsteilnehmer wie Tiere. Die Erfassung und Annotation solcher Spezialfälle sind aufwendig, da sie selten auftreten und nur begrenzt verfügbar sind. Dadurch kann die Zuverlässigkeit und Sicherheit autonomer Systeme in realen Verkehrssituationen beeinträchtigt werden.

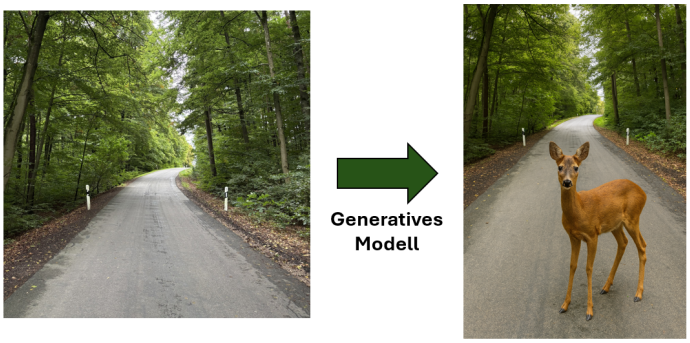

Abbildung 1.1.: Links Originalaufnahme der Waldstraße, rechts mit OpenAI DALL-E3 [13] synthetisch ergänztes Reh zur Simulation seltener Verkehrssituationen.

Eine mögliche Lösung für die eingeschränkte Datenverfügbarkeit in seltenen oder komplexen Szenarien ist der Einsatz generativer KI, wie in Abbildung 1.1 gezeigt. Generative Modelle erzeugen synthetische Daten, die die Verteilung der realen Daten möglichst genau abbilden [14]. In den vergangenen Jahren hat die Entwicklung generativer KI erhebliche Fortschritte gemacht, insbesondere im Bereich der Sprach- und Bildgenerierung [15]. Moderne Modelle sind in der Lage, Bilder und Videos zu erzeugen, die für menschliche Betrachter visuell kaum von echten Aufnahmen zu unterscheiden sind [16]. Ein weiterer Vorteil generativer Modelle besteht in der theoretisch unbegrenzten Datengenerierung. Der Generierungsprozess neuer Datenpunkte wird dabei auch als Sampling-Prozess bezeichnet. Er kann gezielt durch vorgegebene Bedingungen gesteuert werden, sodass Daten aus einer bestimmten Teilverteilung der zugrundeliegenden Datenverteilung generiert werden können. Dies ermöglicht eine kontrollierte und kontextabhängige Datengenerierung, die primär in Bereichen wie der konditionalen Text- oder Bildsynthese von Bedeutung ist [17, 18].

Bevor die generierten Daten für das Training klassifizierender Vorhersagemodelle

verwendet werden, muss ihre Qualität sichergestellt werden [19]. Hierfür kommen verschiedene Evaluierungsmetriken zum Einsatz. Es existieren verteilungsbasierte und instanzbasierte Metriken. Verteilungsbasierte Metriken messen die Distanz zwischen der Verteilung, der realen und generierten Daten [20]. Instanzbasierte Metriken hingegen bewerten die Qualität einzelner Datenpunkte [21]. Die Anwendung einer Metrik zur Validierung synthetischer Daten ist nur gerechtfertigt, wenn ihre Fähigkeit, reale und generierte Daten korrekt zu vergleichen, nachgewiesen wurde [14, 16]. Für Bilddaten kann dieser Nachweis durch menschliche Wahrnehmungsurteile erbracht werden. Dabei gilt eine Metrik als geeignet, wenn ihre Bewertungen mit den Urteilen von Menschen konsistent sind [21–23]. Viele Sensordaten lassen sich jedoch nicht zuverlässig durch Menschen beurteilen, da komplexe oder hochdimensionale Signale schwer interpretierbar sind [21]. Um für diese Sensordaten dennoch eine belastbare Evaluierung der Metriken sicherzustellen, können die Metriken auf Daten, bei denen die Verteilung bekannt ist, ausgewertet werden. Dadurch ist eine objektive Bewertung der Ähnlichkeit möglich, die als Referenzmaß herangezogen werden kann [24]. Die Verteilungen können zum Beispiel durch parametrisch gesteuerte Generatoren erzeugt werden. Die erzeugten Verteilungen hängen von den Parametern des Generators ab. Dadurch lässt sich überprüfen, ob die Metrik Datensätze, die von Generatoren mit ähnlichen Parameterkonfigurationen erzeugt wurden, als ähnlich bewertet.

Während sich viele bestehende Arbeiten auf Metriken fokussieren, die synthetische Bilddaten bewerten [25–28], stellen Radar-Daten eine besondere Herausforderung dar. Sie unterscheiden sich in Struktur, Auflösung und Semantik von visuellen Daten. Daher ist es notwendig zu untersuchen, inwieweit etablierte Metriken aus der Bildverarbeitung wie die FID, auf Radar-Daten übertragbar sind.

Eine weitere zentrale Frage ist, welche generativen Modelle für die Erzeugung synthetischer Radar-Daten geeignet sind. Da sich Radar-Daten strukturell von visuellen Daten unterscheiden, stellen Radar-Daten besondere Anforderungen an generative Modelle. Viele etablierte generative Ansätze wurden primär für visuelle Daten entwickelt und evaluiert [29–32]. Ein Beispiel hierfür sind Diffusionsmodelle, die sich in der Bildgenerierung als besonders leistungsfähige Modellklasse etabliert haben und dort den aktuellen Stand der Forschung repräsentieren.

Vor diesem Hintergrund wird in dieser Arbeit untersucht, ob Diffusionsmodelle auch für die Generierung von Radar-Daten geeignet sind. Im Fokus steht dabei die Frage, ob Diffusionsmodelle Daten erzeugen können, die die charakteristischen Eigenschaften von Radar-Daten abbilden. Zwar existieren bereits erste Arbeiten, die Diffusionsmodelle zur Generierung oder Verbesserung von Radar-Daten einsetzen [33–36], diese sind jedoch anwendungsgetrieben und bewerten die generierten Daten anhand aufgabenspezifischer oder allgemeiner Metriken. In dieser Arbeit wird die Qualität der generierten Radar-Daten hingegen explizit distributionsbasiert mithilfe der zuvor validierten FRD bewertet.

1.2. Zielsetzung und Forschungsfrage

Diese Arbeit verfolgt zwei Ziele. Erstens wird die FRD als Distanzmaß zur Bewertung synthetischer Radar-Daten empirisch evaluiert. Zweitens wird untersucht, inwieweit sich Diffusionsmodelle zur Generierung synthetischer Radar-Daten eignen.

Die FRD wurde entwickelt, um qualitative Unterschiede zwischen realen und generierten Radar-Daten messbar zu machen [37]. Sie orientiert sich konzeptionell an der FID, ist jedoch gezielt an die strukturellen Eigenschaften von Radar-Daten angepasst. Zur Evaluierung der FRD werden zunächst bestehende interne Vorarbeiten repliziert und durch zusätzliche Experimente erweitert. Die Evaluation erfolgt in einer eigens entwickelten Python-basierten Evaluationsumgebung. Ziel ist es, zu überprüfen, ob die FRD in synthetischen, parametrisch kontrollierten Radar-Szenarien, konsistente, erwartungskonforme und reproduzierbare Bewertungen liefert. Damit wird die Eignung der FRD als Qualitätsmaß für Radar-Daten empirisch belegt.

Aufbauend auf dieser Validierung wird im zweiten Teil dieser Arbeit untersucht, ob Diffusionsmodelle in der Lage sind, realistische Radar-Daten zu generieren. Da Radar-Daten eine dünnbesetzte und verrauschte Struktur aufweisen und sich somit von Bilddaten unterscheiden, ist eine Übertragung von Diffusionsmodellen auf Radar-Daten nicht ohne Weiteres gegeben. Die Bewertung der durch die Diffusionsmodelle generierten Daten erfolgt anhand der zuvor validierten FRD. Darüber hinaus wird analysiert, wie sich die Modellkapazität der eingesetzten U-Net-basierten Diffusionsmodelle auf den Trainingsverlauf auswirkt. Im Fokus stehen dabei sowohl die Konvergenzgeschwindigkeit als auch das erreichbare Endniveau der FRD. Ziel ist es, Zusammenhänge zwischen Modellgröße und resultierender Sample-Qualität zu untersuchen.

Forschungsfragen

Die Forschungsfragen sind entlang der beiden Zielsetzungen dieser Arbeit strukturiert.

- **RQ1 – Validität der FRD als Qualitätsmaß:**
  Liefert die FRD in synthetischen, parametrisch kontrollierten Radar-Szenarien konsistente, reproduzierbare und erwartungskonforme Bewertungen?
- **RQ2 – Generative Eignung von Diffusionsmodellen:**
  Sind Diffusionsmodelle in der Lage, die statistischen und strukturellen Eigenschaften parametrisch erzeugter Radar-Punktwolken zu reproduzieren?
- **RQ3 – Abbildung der Trainingsdynamik:**
  Spiegelt die FRD die qualitative Entwicklung der generierten Radar-Daten während des Trainingsprozesses eines Diffusionsmodells zuverlässig wider?

- **RQ4 – Einfluss der Modellkapazität:**
  Wie beeinflusst die Modellgröße eines U-Net-basierten Diffusionsmodells die Konvergenzgeschwindigkeit und das erreichbare Endniveau der FRD?

1.3. Umfeld der Arbeit

Die Bachelorarbeit erfolgt im Rahmen meines dualen Studiums bei AUMOVIO Germany GmbH am Standort Regensburg, in der ich in der Abteilung A OT INN AIE CORE tätig bin. Mein fachlicher Betreuer, Dr. Jörg Reichardt, leitet dort das Public-Private-Partnership (PPP)-Projekt nxtAIM [38], das den Rahmen für diese Arbeit vorgibt. PPP-Projekte sind Projekte, die sich durch eine enge Zusammenarbeit zwischen öffentlichen Einrichtungen und privatwirtschaftlichen Unternehmen auszeichnen. Die Projekte verfolgen das Ziel, innovative Lösungen für gesellschaftlich relevante Fragestellungen zu entwickeln. NxtAIM ist ein Projekt, das vom Bundesministerium für Wirtschaft und Energie gefördert wird und das Ziel verfolgt, Methoden der Generativen KI für den Einsatz im Bereich des autonomen Fahrens anzupassen. Dafür werden generative Modelle für die im Fahrzeug erhobenen Sensordaten entwickelt [39]. Im Zuge des Projektes haben sich Jonas Neuhöfer und Dr. Jörg Reichardt [37] mit der Frage beschäftigt, wie man die Qualität von generativen Modellen für Radar-Daten messen kann. Meine Bachelorarbeit schließt an die Überlegungen und Ergebnisse ihrer Arbeit an.

1.4. Aufbau der Arbeit

Die Arbeit ist in sechs Kapitel gegliedert. Nach der Einleitung führt das 2. Kapitel die für das Verständnis der Arbeit notwendigen theoretischen Grundlagen ein. Dazu zählen die Eigenschaften und Repräsentationsformen von Radar-Daten, zentrale Konzepte generativer Modelle mit Fokus auf Diffusionsmodelle sowie etablierte Metriken zur Bewertung synthetischer Daten. Ergänzend werden die mathematischen Grundlagen zufälliger Projektionen sowie die Schätzung von Mittelwert und Kovarianz behandelt, die für die spätere Definition der FRD erforderlich sind. Das 3. Kapitel beschreibt die in dieser Arbeit verwendete Methodik. Zunächst wird die FRD als Distanzmaß für synthetische Radar-Punktwolken vorgestellt und ihre Berechnung beschrieben. Anschließend werden parametrisierte Pseudoradar-Generatoren als kontrollierte Datenquelle vorgestellt. Im 4. Kapitel werden die durchgeführten Experimente und deren Ergebnisse präsentiert. Zunächst wird dafür die FRD in kontrollierten Szenarien validiert. Dafür wird ihr Grenzverhalten bei identischen Verteilungen sowie ihre Sensitivität gegenüber gezielten Verteilungsänderungen untersucht. Darauf aufbauend wird die FRD zur Bewertung eines U-Net-basierten Diffusionsmodells eingesetzt, wobei speziell die Trainingsdynamik und die Qualität der generierten Radar-Punktwolken analysiert werden. Das 5. Kapitel ordnet die Ergebnisse im Kontext der Forschungsfragen ein

und diskutiert die Befunde. Dabei werden sowohl Stärken als auch Limitationen der Arbeit vorgestellt. Abschließend fasst das 6. Kapitel die zentralen Erkenntnisse der Arbeit zusammen und gibt einen Ausblick auf mögliche Weiterentwicklungen und zukünftige Forschungsrichtungen.

2. Theoretische Grundlagen

Im folgenden Kapitel werden die theoretischen Konzepte und mathematischen Grundlagen erläutert, die für das Verständnis dieser Arbeit relevant sind.

2.1. Radar

Radar bezeichnet ein Messverfahren zur Erkennung und Distanzbestimmung von Objekten mithilfe elektromagnetischer Wellen. Ein Radarsystem besteht aus einem Sender (Transmitter) und einem Empfänger (Receiver). Der Transmitter sendet in regelmäßigen Intervallen elektromagnetische Wellen im Radio- oder Mikrowellenbereich mit definierter Frequenz aus. Treffen die ausgesendeten Wellen auf ein Objekt, werden sie teilweise reflektiert und gelangen zurück zum Radar, wo sie vom Receiver erfasst werden. Im Signalverarbeitungssubsystem erfolgt die Analyse der Laufzeit zwischen Senden und Empfangen. Dadurch kann die relative Distanz zum Objekt bestimmt werden. Zusätzlich kann die Frequenzverschiebung des reflektierten Signals ausgewertet werden, um mithilfe des Doppler-Effektes die Relativgeschwindigkeit des Objekts zu berechnen [40, 41].

Um zusätzlich den horizontalen und vertikalen Winkel eines detektierten Objektes, relativ zum Radar zu erhalten, können mehrere Radarantennen in einem Array angeordnet werden. Durch die Anordnung in einem Array werden Objekte gleichzeitig von mehreren Empfangsantennen detektiert. Dabei unterscheidet sich die Laufzeit der Reflexion abhängig von der Position der Antenne im Array. Durch das Vergleichen der Laufzeiten von horizontal angeordneten Antennen lässt sich der Horizontalwinkel bestimmen, der auch als Azimutwinkel bezeichnet wird. Das Vergleichen der Laufzeiten von vertikal angeordneten Antennen lässt sich der Vertikalwinkel ermitteln [42]. Mithilfe der Informationen über Vertikalwinkel, Horizontalwinkel und der Distanz eines Objektes relativ zum Sensor kann seine Position (x, y, z) in einem dreidimensionalen Raum bestimmt werden [43]. Ein weiterer Vorteil, der durch die Verwendung mehrerer Antennen entsteht, ist die genauere Bestimmung der relativen Distanz eines Objekts zum Radar [44].

Neben den grundlegenden Funktionsprinzipien beeinflussen Störeinflüsse Radarsysteme. Sie wirken sich auf die Genauigkeit und Zuverlässigkeit der detektierten Objekte aus. Störeinflüsse entstehen beispielsweise durch ungewollte Reflexionen, die zur Detektion sogenannter Clutter-Punkte führen. Eine ungewollte Reflexion ist die Mehrwegeausbreitung. Sie entsteht, wenn eine ausgesendete Radarwelle nicht nur direkt vom Zielobjekt reflektiert wird, sondern zusätzlich von einem

weiteren Objekt oder einer Oberfläche wie dem Boden oder Gebäuden. Diese mehrfach reflektierten Signale gelangen zeitverzögert zurück zum Radar und können zu Fehldetektionen oder falschen Positionsangaben führen. Auch elektronische Geräte, die ebenfalls elektromagnetische Wellen aussenden, können zu Störreflexionen führen. Diese unerwünschten Signale erschweren die Zielerkennung und erfordern eine gezielte Clutter-Unterdrückung durch geeignete Signalverarbeitungsalgorithmen [45].

Trotz dieser Herausforderungen bietet Radar gegenüber optischen Verfahren einige Vorteile. Im Gegensatz zu LiDAR und Kameras, die beide im oder nahe dem sichtbaren Spektrum arbeiten, nutzt Radar elektromagnetische Wellen außerhalb dieses Bereichs. Dadurch ist Radar deutlich robuster gegenüber Umwelteinflüssen wie Sonnenlicht, Staub sowie Wetterbedingungen wie Regen, Schnee und Nebel [45–47]. Ein weiterer Vorteil von Radar ist die direkte Messung der Objektentfernung. Zudem sind Radarsysteme im Vergleich zu LiDAR-Systemen deutlich kostengünstiger und kleiner, wodurch sie einen entscheidenden Vorteil für den industriellen Einsatz bieten [48].

Ein weiterer Nachteil von Radar ist jedoch die vergleichsweise geringe räumliche Auflösung. Diese liegt in der Regel unter der von LiDAR-Systemen und wirkt sich insbesondere auf die präzise Erfassung von Objektformen und -kanten aus [49].

Die beschriebenen Eigenschaften machen Radar zu einer Schlüsseltechnologie in zahlreichen Anwendungsfeldern. Ein Beispiel ist die Luftraumüberwachung, bei der Radar zur Identifikation von Flugzeugen und zur Vermeidung von Kollisionen eingesetzt wird. Ähnlich wird Radar in der Schifffahrt genutzt, um Zusammenstöße mit anderen Schiffen und anderen schwimmenden Objekten zu verhindern. Darüber hinaus findet Radar auch in der Raumfahrt Anwendung, beispielsweise beim Andocken von Raumfahrzeugen oder bei Landungen auf dem Mond [41]. Neben diesen Bereichen wird Radar auch in der Meteorologie zur Niederschlagsmessung und Sturmüberwachung eingesetzt, in den Geowissenschaften für Erdbeobachtung und topografische Messungen, sowie im militärischen und sicherheitskritischen Umfeld zur Zielerfassung und Überwachung. Diese Beispiele verdeutlichen das breite Einsatzgebiet von Radar in sicherheitsrelevanten, industriellen und wissenschaftlichen Anwendungen. Neben den genannten Einsatzgebieten spielt Radar auch im Automobilsektor eine zentrale Rolle, insbesondere im Kontext moderner Fahrerassistenzsysteme und autonomer Fahrzeuge [42].

2.1.1. Radar im Auto

Im Automobilbereich hat sich Radar als robuste und kosteneffiziente Sensortechnologie für Fahrerassistenzsysteme etabliert. Automotive-Radarsensoren arbeiten typischerweise im Millimeterwellenbereich, speziell im Frequenzband um 77 GHz, und ermöglichen die zuverlässige Detektion von Objekten über große Entfernungen. Aufgrund der gleichzeitigen Messung von Distanz, Relativgeschwindigkeit

und Winkelposition eignet sich Radar besonders für dynamische Verkehrsszenarien mit mehreren bewegten Objekten. Radarsensoren bilden somit eine zentrale Grundlage für sicherheitskritische Funktionen wie adaptive Geschwindigkeitsregelungen, Kollisionswarnsysteme und automatische Notbremsassistenten [50].

2.1.2. Datenstruktur von Radar-Daten

Radar-Daten können nach ihrer Verarbeitung in unterschiedlichen Datenformaten vorliegen. Ein weit verbreitetes Format ist der sogenannte Range-Azimut-Doppler-Tensor (RAD-Tensor). Dabei handelt es sich um eine dreidimensionale Repräsentation der Radar-Daten. Sie enthält die Informationen über die Entfernung zu einem detektierten Objekt (Range), den horizontalen Winkel relativ zum Sensor (Azimut) sowie die relative Geschwindigkeit des Objekts zum Sensor (Doppler). Falls zusätzlich Daten zum Vertikalwinkel vorliegen und mehrere Punkte mit unterschiedlichen Vertikalwinkeln in dieselbe Zelle fallen, wird deren Mittelwert berechnet, um eine konsistente 3D-Darstellung zu erhalten [42]. Die Darstellung von Radar-Daten als Tensor bzw. Datenwürfels bietet den Vorteil einer hohen Informationsdichte und ermöglicht eine präzise Lokalisierung und Bewegungserkennung von Objekten. Gleichzeitig stellt die hohe Dimensionalität und Dichte der Daten eine Herausforderung für die Weiterverarbeitung mit Modellen des maschinellen Lernens dar, da sie mit einem erheblichen Rechenaufwand verbunden ist [51].

Wenn bei der Verarbeitung von Radar-Daten ausschließlich die ersten beiden Dimensionen des RAD-Tensors betrachtet werden und die Doppler-Dimension dabei komprimiert oder ignoriert wird, entsteht eine sogenannte Range-Azimut-Heatmap. Diese Darstellung entspricht einem zweidimensionalen Bild aus der Vogelperspektive, indem die vom Radar erfassten Objekte als Intensitätswerte visualisiert werden. Der Vorteil dieses Formats liegt in der erheblichen Reduktion des Rechenaufwands, da die Datenmenge im Vergleich zum vollständigen RAD-Tensor deutlich geringer ist. Allerdings geht diese Vereinfachung mit dem Verlust der Doppler-Information einher, da die relative Geschwindigkeit der Objekte zum Sensor nicht mehr direkt aus der Darstellung abgeleitet werden kann [42].

Ein weiteres etabliertes Datenformat in der Radarsignalverarbeitung ist die Radar-Punktwolke. Ein Beispiel aus dem Automobilbereich ist in Abbildung 2.1 dargestellt. Bei der Erzeugung einer Radar-Punktwolke werden aus den Rohdaten des Radarsensors die Zellen selektiert, die ein signifikantes reflektiertes Signal aufweisen. Jede dieser Zellen wird anschließend als diskreter Messpunkt in einem mehrdimensionalen Merkmalsraum repräsentiert. Die Definition der Achsen dieses Merkmalsraums kann auf unterschiedliche Weise erfolgen. Eine Möglichkeit besteht darin, die Punkte im Sensorkoordinatensystem darzustellen, beispielsweise durch die Merkmale Distanz, Azimutwinkel und Relativgeschwindigkeit. In diesem Fall ist die Radar-Punktwolke strukturell an die ursprüngliche Sensorrepräsentation angelehnt. Alternativ kann die Punktwolke in einen kartesischen dreidimensionalen Raum überführt werden, indem Winkel- und Distanzinformationen in x-, y- und

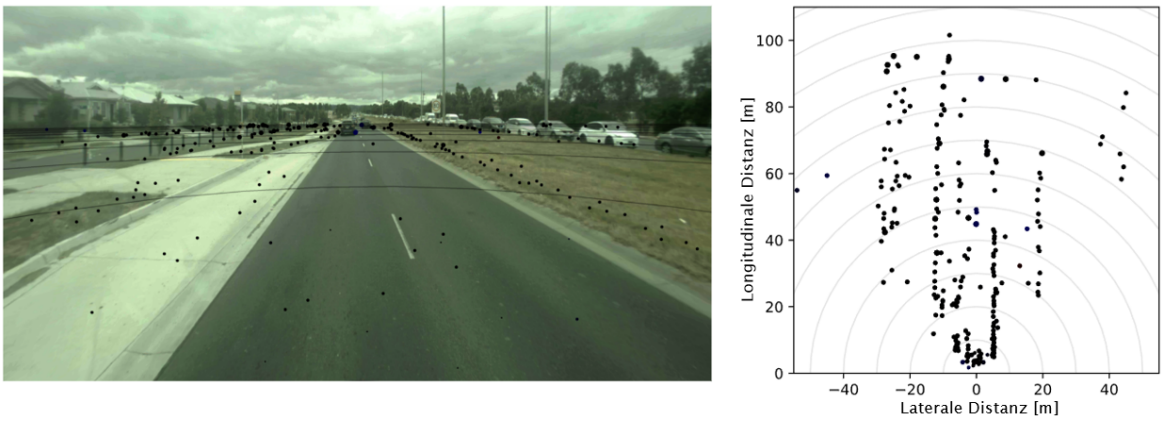

Abbildung 2.1.: Links: Kameraaufnahme des Fahrzeugs. Rechts: korrespondierende Radar-Punktwolke in Vogelperspektive, dargestellt mit longitudinaler Distanz (y-Achse) und lateraler Distanz (x-Achse). Die Punktwolke zeigt linienförmige Strukturen, die Straße, Zaun und parkende Fahrzeuge repräsentieren. Vereinzelte Punkte ohne Struktur sind auf Störeinflüsse zurückzuführen. ©AUMOVIO

z-Koordinaten transformiert werden. Diese Darstellung erlaubt eine räumliche Interpretation der detektierten Objekte, die mit der Perspektive visueller Sensoren wie Kameras kompatibel ist [52]. Zusätzlich zu den räumlichen Koordinaten können jedem Punkt weitere radarspezifische Messgrößen zugeordnet werden, etwa die Relativgeschwindigkeit, der vertikale Winkel oder die Reflexionsstärke [42]. Ein wesentlicher Vorteil der Radar-Punktwolke besteht darin, dass sie eine deutliche Reduktion der Datenmenge gegenüber der ursprünglichen Sensordarstellung ermöglicht, ohne dabei relevante Informationen zu verlieren [42].

2.2. Künstliche Intelligenz, Maschinelles Lernen und Generative Künstliche Intelligenz

Während KI als Oberbegriff für Systeme dient, die in ihrer Umwelt intelligentes Verhalten nachahmen, konzentriert sich der Teilbereich des ML darauf, Algorithmen zu entwickeln, die es Computern ermöglichen, autonome Entscheidungen basierend auf Daten zu treffen [53]. Im Gegensatz zu klassischen Algorithmen basieren diese Entscheidungen nicht ausschließlich auf ihrer Programmierung, sondern zusätzlich auf den Daten, auf die die Algorithmen angewendet wurden.

Eine gängige Möglichkeit, ML in einzelne Kategorien zu unterteilen, besteht darin, Modelle nach der Art ihres Lernprozesses einzuordnen. Mithilfe dieser Perspektive lässt sich ML in drei grundlegende Lernparadigmen unterteilen. Das erste Lernparadigma ist das überwachte Lernen. Beim überwachten Lernen sind Paare aus Eingabedaten $x \in \mathcal{X}$ und Zielwerten $y \in \mathcal{Y}$ gegeben. Sie werden genutzt, um eine Funktion $y \approx f(\theta, x)$ zu lernen, die die unbekannte wahre Zielabbildung

$f^*(x)$, möglichst gut approximiert. Die Funktion bildet somit Eingabedaten x auf Zielwerte y ab. Das Ziel ist es, die Parameter θ so zu optimieren, dass der Fehler $\mathcal{L}(y, f(\theta, x))$ minimal wird. Somit werden beim überwachten Lernen Funktionen gelernt, die Datenpunkte aus einem definierten Eingaberaum auf Datenpunkte im Ausgaberaum abbilden [54]. Dieser Ansatz wird mit der Annahme begründet, dass es eine unbekannte Verteilung $p(x, y)$ gibt, aus der die Daten stammen, sodass die bedingte Verteilung $p(y \mid x)$ gelernt werden kann.

Im Gegensatz zum überwachten Lernen stehen beim unüberwachten Lernen lediglich Datenpunkte $x \in \mathcal{X}$ ohne zugehörige Labels zur Verfügung [55]. Ziel beim unüberwachten Lernen ist es, die zugrundeliegende Struktur beziehungsweise die Verteilung der Eingabedaten x zu modellieren. Formal gilt:

$$p_\lambda(x) \approx p_{\text{daten}}(x). \tag{2.1}$$

Die Modellierung der Verteilung der Eingabedaten kann dazu genutzt werden, verborgene Strukturen wie Cluster innerhalb der Daten zu identifizieren. Durch die Identifikation struktureller Zusammenhänge lassen sich redundante oder wenig informative Merkmale in den Daten erkennen. Auf dieser Grundlage kann eine Reduktion der Dimensionalität vorgenommen werden, wodurch die Daten auf ihre wesentlichen Merkmale reduziert werden.

Die dritte Kategorie des ML ist das bestärkende Lernen. Hier werden Weltzustände auf Aktionen abgebildet, um ein numerisches Belohnungssignal zu maximieren. Die optimalen Aktionen sind dabei nicht bekannt, sondern müssen iterativ durch Interaktion mit der Umwelt erlernt werden. Dabei ändert sich der Zustand der Umwelt in Abhängigkeit von den gewählten Aktionen kontinuierlich. Ziel ist es daher nicht nur, die Belohnung im nächsten Schritt zu maximieren, sondern eine Strategie zu entwickeln, die den erwarteten Gesamtertrag über viele Schritte hinweg optimiert [56]. Anwendung findet bestärkendes Lernen vor allem in Spielen wie Schach, aber auch in Bereichen der Robotik.

2.2.1. Von diskriminativen zu generativen Ansätzen der KI

Die Einteilung von Modellen des maschinellen Lernens in überwachte, unüberwachte und bestärkende Lernverfahren beschreibt die Art des Trainingsprozesses, jedoch nicht das Ziel, das ein Modell verfolgt. Dadurch lassen sich jedoch Modelle, die neue Daten generieren, nicht von Modellen, die Daten Labels zuordnen, unterscheiden.

Der Grund hierfür liegt darin, dass sowohl diskriminative als auch generative Modelle mit unterschiedlichen Lernparadigmen trainiert werden können. Beispielsweise werden große Sprachmodelle, deren Ziel es ist neue Textsequenzen zu generieren mithilfe von selbstüberwachtem Lernen trainiert. Beim selbstüberwachten Lernen stammen die Trainingsdaten X und die dazugehörigen Zielwerte

Y vollständig aus der Datenverteilung selbst. Im Fall von Sprachmodellen äußert sich dies typischerweise in der Vorhersage des nächsten Tokens auf Basis des bisherigen Kontexts [57]. Die Eingabe x besteht dabei aus dem bisherigen Kontext, während das Ziel y als das nächste Element der Sequenz direkt aus den Rohdaten abgeleitet wird. Da keine extern annotierten Labels benötigt werden, gilt selbstüberwachtes Lernen als Unterform des unüberwachten Lernens. Auch andere generative Modelle wie Variational Autoencoder [30] oder Diffusionsmodelle [31] werden ohne explizite Zielvorgaben trainiert. Sie lernen stattdessen eine Approximation der zugrundeliegenden Datenverteilung.

Neben selbstüberwachtem Lernen kommt bei generativen Modellen auch bestärkendes Lernen zum Einsatz. Ein Beispiel ist Reinforcement Learning from Human Feedback, bei dem ein Belohnungsmodell auf Basis menschlicher Präferenzdaten trainiert wird, das anschließend das Verhalten des Modells steuert [58]. Auch überwachtes Lernen wird für das Training von Modelle genutzt. Beispielsweise im Rahmen des Supervised-Fine-Tunings. Dabei werden generative Modelle, mit von Menschen erstellten Eingabe-Ausgabe-Paaren trainiert, um Fähigkeiten wie Instruktionsbefolgung oder der Verwendung eines bestimmten Antwortstils zu erlernen [59].

Diese Beispiele verdeutlichen, dass die Einordnung von Modellen in Lernparadigmen nicht dafür geeignet ist, Modelle, die neue Daten generieren, von Modellen zu unterscheiden, die neue Daten klassifizieren. Beide Modelle können dieselben Lernparadigmen benutzen und dadurch ähnliche Optimierungsziele verfolgen. Somit lassen sie sich nicht allein auf der Ebene des Trainingsprozesses unterscheiden. Die Abgrenzung von generativen Modellen erfolgt daher nicht über die Formulierung des Trainingsprozesses, sondern über das zugrundeliegende Ziel des Modells. Dabei unterscheidet man zwischen diskriminativen Modellen, die Daten klassifizieren und generativen Modellen, die neue Datenpunkte erzeugen [60].

2.3. Generative Modelle

Generative Modelle verfolgen das zentrale Ziel, neue Datenpunkte zu erzeugen, die statistisch konsistent mit einer gegebenen Trainingsdatenverteilung sind. Zu diesem Zweck lernen sie implizit oder explizit eine Approximation der zugrundeliegenden Datenverteilung, aus der anschließend neue Stichproben generiert werden können. Dadurch lassen sie sich in zwei verschiedene Kategorien unterteilen. Implizite generative Modelle lernen eine Abbildung, die eine einfache, bekannte Ausgangsverteilung in eine komplexe Datenverteilung überführt. Der Ansatz lässt sich formal wie folgt ausdrücken:

$$z \sim p(z), \quad x = g_\phi(z), \qquad p_g \approx p_{\text{daten}}. \tag{2.2}$$

Dabei werden die Parameter ϕ so gewählt, dass ein geeignetes Abstands- oder Divergenzmaß $D(p_{\text{daten}}, p_g)$ minimiert wird, welches den Unterschied zwischen der Verteilung der generierten Daten und der empirischen Datenverteilung quantifiziert. Somit wird die Trainingsdatenverteilung nicht explizit modelliert. Explizite generative Modelle modellieren die Wahrscheinlichkeitsverteilung der Daten direkt. Die Modellparameter werden so optimiert, dass die Trainingsdaten unter dem Modell eine hohe Wahrscheinlichkeit erhalten. Dies entspricht einer Likelihood basierten Optimierung, etwa in Form der Maximierung der Likelihood oder einer von ihr abgeleiteten Zielfunktion.

Auf Basis dieser unterschiedlichen Modellierungsprinzipien haben sich verschiedene Klassen generativer Modelle etabliert [61–63]. Die erste Klasse generativer Modelle bilden Generative Adversarial Network (GAN)s, die 2014 von Goodfellow et al. vorgestellt wurden [29]. GANs bestehen aus zwei gegeneinander spielenden neuronalen Netzwerken, dem Generator, der neue Datenproben erzeugt, und dem Diskriminator, der zwischen echten und generierten Daten unterscheidet. Das Training erfolgt in Form eines Minimax-Spiels. Während der Generator versucht, den Diskriminator zu täuschen und damit möglichst realistische Daten zu erzeugen, verbessert der Diskriminator gleichzeitig seine Fähigkeit, generierte von echten Daten zu unterscheiden. Durch diese Wechselwirkung lernen beide Netzwerke kontinuierlich und voneinander. Das ermöglicht dem Generator, die zugrundeliegende Datenverteilung komplexer Datenstrukturen nachzubilden, ohne die explizite Wahrscheinlichkeitsdichte der Daten modellieren zu müssen [29, 61–63].

Eine weitere Klasse generativer Modelle bilden Variational Autoencoder (VAE)s. Sie wurden erstmals von Kingma und Welling 2014 in [30] vorgestellt. VAEs basieren auf der Idee, dass komplexe Datenverteilungen über latente Variablen modelliert werden können. Sie bestehen aus einem Encoder, der die Eingabedaten auf eine niedrigdimensionale latente Verteilung abbildet, und einem Decoder, der diese Verteilung zurück auf die Datenpunkte abbildet. Das Training verfolgt zwei Ziele. Einerseits sollen die Eingabedaten rekonstruiert werden, indem der Encoder sie in den latenten Raum überführt und der Decoder versucht, sie möglichst genau wiederherzustellen. Andererseits wird die Struktur des latenten Raums reguliert. Dabei erzwingt die Regularisierung, dass die vom Encoder gelernte latente Verteilung mit einer einfachen, in der Regel normalverteilten Grundverteilung übereinstimmt. Dadurch können nach dem Training neue Daten erzeugt werden, indem Daten aus einer Normalverteilung gezogen werden und anschließend durch den Decoder in den Datenraum transformiert werden. Auch bei VAEs wird die Datenverteilung nicht explizit modelliert. Stattdessen wird eine Abbildung gelernt, die eine multivariate Normalverteilung so transformiert, dass die dadurch erzeugten Daten der Verteilung der Trainingsdaten entsprechen [30, 61–63].

Autoregressive Modelle sind eine weitere Klasse generativer Modelle. Sie basieren auf der Zerlegung der Wahrscheinlichkeitsverteilung einer Datenmenge in ein Produkt bedingter Wahrscheinlichkeiten [61, 62, 64–66]. Für einen Datenpunkt

$x = (x_1, x_2, \ldots, x_n)$ gilt dabei:

$$p(x) = \prod_{i=1}^{n} p(x_i \mid x_1, \ldots, x_{i-1}). \tag{2.3}$$

Die Wahrscheinlichkeit des nächsten Sequenzelements wird bedingt auf die zuvor beobachteten Elemente modelliert. Das Modell lernt, diese bedingten Wahrscheinlichkeiten zu approximieren. Beim Training wird die Likelihood der Trainingsdaten maximiert, was der Optimierung dieser bedingten Wahrscheinlichkeiten entspricht. Nach dem Training können neue Daten durch sequenzielles Sampling generiert werden, bei dem jeweils aus der bedingten Verteilung des nächsten Elements gesampelt wird. Dadurch sind Autoregressive Modelle besonders leistungsfähig in Szenarien, in denen die Reihenfolge der Daten entscheidend ist und langfristige Abhängigkeiten berücksichtigt werden müssen. Beispiele dafür sind die Text- und Sprachgenerierung [64].

Eine weitere Modellklasse stellen Energy-Based-Models dar. Sie beschreiben Datenverteilungen über eine Energiefunktion $E_\theta(x)$, die jedem möglichen Datenpunkt einen skalaren Energiewert zuordnet [62, 65, 66]. Eine Energiefunktion bildet Datenpunkte mit hoher Wahrscheinlichkeit auf niedrige Energiewerte ab und Datenpunkte mit niedriger Wahrscheinlichkeit auf hohe Energiewerte. Das Training von Energy-Based-Models zielt darauf ab, die Energiefunktion so zu parametrieren, dass der Energiewert der Trainingsdaten minimiert wird. Ein Vorteil gegenüber Modellen, die die Wahrscheinlichkeitsverteilung der Daten explizit modellieren, besteht darin, dass die Energiefunktion nicht normiert sein muss. Die Erzeugung neuer Datenpunkte erfolgt daher typischerweise nicht direkt, sondern indem iterativ Regionen niedriger Energie erkundet werden [62].

Score-basierte Modelle bilden eine weitere Klasse generativer Modelle. Sie modellieren nicht die Datenverteilung selbst, sondern den Gradienten der Log-Dichte der Datenverteilung. Somit lernen sie ausschließlich die Richtung zunehmender Wahrscheinlichkeit, die durch den Score $s(x) = \nabla_x \ln p(x)$ beschrieben wird [62]. Das Modell lernt, indem es versucht, die Gradienten der Log-Dichte der Trainingsdaten nachzubilden [62]. Nach dem Training können neue Daten durch iterative Verfahren erzeugt werden. Ausgehend von einem Punkt, der zufällig im Raum erzeugt wurde, wird dieser schrittweise entlang des gelernten Scores in Richtung der Datenverteilung bewegt [62]. In der praktischen Anwendung werden score-basierte Modelle heute nahezu ausschließlich in Form von Diffusionsmodellen realisiert, die eine stabile Optimierung und effiziente Generierung ermöglichen [31].

2.3.1. Diffusionsmodelle

Diffusionsmodelle haben sich in den letzten Jahren als eine der leistungsfähigsten Methoden für die Bildgenerierung etabliert [67]. Sie zeichnen sich durch hohe

Stabilität im Training und die Fähigkeit aus, qualitativ hochwertige und vielfältige Bilder zu erzeugen [68]. Diffusionsmodelle wurden erstmals von Sohl-Dickstein et al. in [69] als Denoising-Diffusion-Probabilistic-Models (DDPM) vorgestellt. Sie werden durch eine parametrisierte Markov-Kette beschrieben. Die Markov-Kette transformiert eine bekannte Verteilung schrittweise in die gewünschte Zielverteilung. Eine Markov-Kette ist dabei eine Sequenz von Zuständen, bei der jeder Folgezustand nur vom aktuellen Zustand abhängt [69].

Die Parameter der Markov-Kette werden im Training so optimiert, dass sie die Umkehrung eines Diffusionsprozesses darstellen. Ein Diffusionsprozess ist ein stochastischer Prozess, bei dem ein System mit niedriger Entropie durch sukzessives Hinzufügen von Rauschen in ein System mit hoher Entropie überführt wird [70]. Formal lässt sich der Vorwärtsdiffusionsprozess als Markov-Kette darstellen, bei der über eine feste Anzahl von Schritten jeweils eine kleine Menge an Rauschen zum aktuellen Zustand hinzugefügt wird. Nach einer festgelegten Anzahl an T Schritten ist das ursprüngliche Signal vollständig zerstört und der Endzustand besteht nur noch aus zufälligem Rauschen [31, 62]. Der gesamte Ablauf dieses Vorwärtsprozesses ist anhand eines Beispiels in Abbildung 2.2 dargestellt.

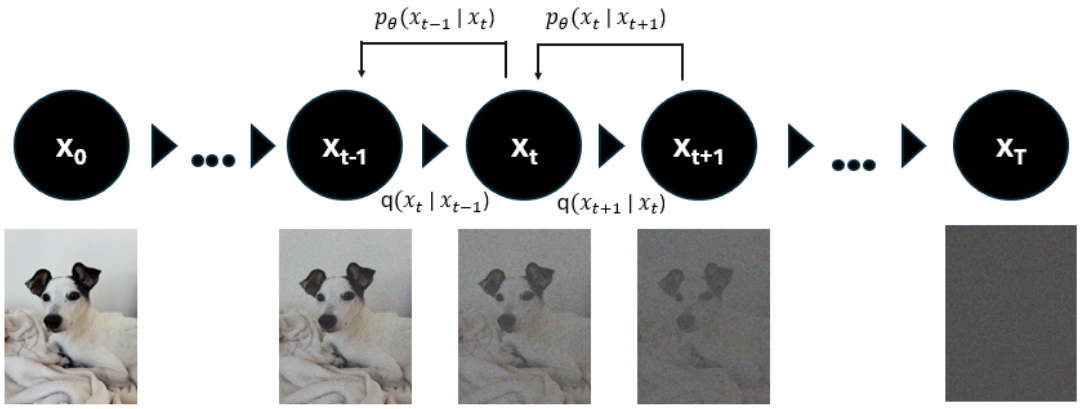

Abbildung 2.2.: Vereinfachte Darstellung des Vorwärtsdiffusionsprozesses und des Rückwärtsprozesses anhand eines Beispiels.

Das Diffusionsmodell lernt die Umkehrung des Vorwärtsprozesses, indem es über viele kleine Schritte hinweg das Hinzufügen von Rauschen rückgängig macht. Dieser Ansatz führt zu einer deutlich stabileren Optimierung als das direkte Lernen einer Abbildung von einer bekannten Ausgangsverteilung auf die Zielverteilung [69]. Da für jede Zielverteilung ein Diffusionsprozess existiert, können Diffusionsmodelle beliebige Verteilungen approximieren [69]. Die Zielverteilung wird implizit über einen parametrisierten Reverse-Diffusionsprozess modelliert, wobei eine Likelihood-Schätzung möglich ist. [63, 69].

In den folgenden Abschnitten wird die mathematischen Formulierungen des Trainings vorgestellt. Anschließend werden verschiedene Sampling-Verfahren und die Architektur des Modells beschrieben.

2.3.1.1. Training

Der Trainingsprozess eines Diffusionsmodells besteht aus dem Vorwärtsdiffusionsprozess und einem Reverse-Diffusion-Prozess. Der Vorwärtsprozess beschreibt dabei einen Gaußschen Diffusionsprozess [71] bei dem jeweils zum aktuellen Zustand Gaußsches Rauschen addiert wird. Dieser Prozess lässt sich als Markov-Kette beschreiben und mathematisch definieren als [31, 68, 69]:

$$q\left(x_{1:T} \mid x_0\right) := \prod_{t=1}^{T} q\left(x_t \mid x_{t-1}\right) \tag{2.4}$$

$$q\left(x_t \mid x_{t-1}\right) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t} \cdot x_{t-1}, \beta_t \mathbf{I}\right). \tag{2.5}$$

Dabei beschreibt die Formel, dass x_t aus einer multivariaten Normalverteilung gezogen wird, die die Mittelwerte $\sqrt{1-\beta_t} \cdot x_{t-1}$ und die über alle multivariaten Verteilungen konstante Kovarianz β_t hat. Wobei β_t die Menge an Rauschen bestimmt, die in jedem Schritt des Vorwärtsprozesses hinzugefügt wird und die gesamte Sequenz von β_t als Noise-Schedule bezeichnet wird. Die β_t werden entweder durch Reparametrisierung gelernt [69, 72] oder als Hyperparameter gewählt [31, 67]. Der Hyperparameter T beschreibt die Anzahl der Diffusionsschritte. In der Praxis wird häufig $T = 1000$ gewählt. Das Ziel des Vorwärtsprozesses ist, nach T Schritten den Zustand x_T in eine wohldefinierte Verteilung zu konvertieren. Dadurch wird der Sampling-Prozess berechenbar [69]. Die Formel 2.5 ist jedoch für das Training ungeeignet, da um ein verrauschtes Bild zum Zeitpunkt $T = t$ zu erhalten, ein kompletter Durchlauf der Markov-Kette bis zum Zeitschritt t nötig wäre. Die Formel lässt sich jedoch mithilfe des Reparametrisierungstricks für Normalverteilungen und anschließendes Ausmultiplizieren vereinfachen zu:

$$q\left(x_t \mid x_0\right) := (\prod_{i=1}^{t} \sqrt{1-\beta_i}) \cdot x_0 + \varepsilon \cdot \sqrt{1 - \prod_{i=1}^{t}(1-\beta_i)} \qquad \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.6}$$

oder wie in wissenschaftlichen Veröffentlichungen üblicher [31, 32, 67, 73] mit $\hat{\alpha}_t = \prod_{i=1}^{t} \alpha_i$ und $\alpha_t = 1 - \beta_t$ als:

$$q\left(x_t \mid x_0\right) := \mathcal{N}\Big(x_t; \sqrt{\hat{\alpha}_t} \cdot x_0, (1-\hat{\alpha}_t)\mathbf{I}\Big). \tag{2.7}$$

Diese Gleichung ermöglicht es, den Zustand x_t für einen Diffusionsschritt t direkt zu berechnen [67] und beschleunigt somit den Vorwärtsprozess.

Für den Reverse-Prozess stellten Sohl-Dickstein et al. in [69] fest, dass für kleine β_ts $q\left(x_{t-1} \mid x_t\right)$ auch einer Gaußschen Verteilung folgt. Diese Erkenntnis ist dahingehend entscheidend, da sich der Reverse-Prozess deshalb darauf reduzieren lässt, die Mittelwerte und die Kovarianz des Reverse-Diffusion-Kernels zu

bestimmen [69]. Es gilt [31]:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\big(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\big), \tag{2.8}$$

wobei $\mu_\theta(x_t, t)$ und $\Sigma_\theta(x_t, t)$ jeweils Funktionsapproximatoren sind. Sie lernen die Mittelwerte und Kovarianz für den Reverse-Zeitschritt t vorherzusagen [31, 69]. Die beiden Funktionsapproximatoren können trainiert werden, indem sie die Log-Likelihood der Trainingsdaten maximieren, wobei die Wahrscheinlichkeit für einen Datenpunkt x_0 gegeben ist durch:

$$p_\theta(x_0) = \int p_\theta(x_{0:T})\, dx_{1:T} = \int p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)\, dx_{1:T}. \tag{2.9}$$

Dabei ist $p(x_T)$ der Endzustand des Vorwärtsprozesses und somit normalverteilt. Das Integral ist jedoch nicht berechenbar, da dafür die Wahrscheinlichkeit aller möglichen Sequenzen, die von $p(x_T)$ nach x_0 über die T Diffusionsschritte führt, berechnet werden müsste [69]. Die Wahrscheinlichkeit kann jedoch umgeformt werden zu:

$$p_\theta(x_0) = \int q\big(x_{1:T} \mid x_0\big) \cdot p(x_T) \cdot \prod_{t=1}^{T} \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})}\, dx_{1:T}. \tag{2.10}$$

Diese Wahrscheinlichkeit kann effizient mittels Monte-Carlo-Sampling [74] approximiert werden, indem der Mittelwert über viele Stichproben des Vorwärtsprozesses $q(x_{1:T} \mid x_0)$ berechnet wird [69].

Die Parameter der Approximationsfunktionen $\mu_\theta(x_t, t)$ und $\Sigma_\theta(x_t, t)$ werden optimiert, indem sie die Wahrscheinlichkeit der Trainingsdaten unter dem Reverse-Diffusionsprozess maximieren. Dies entspricht der Maximierung der Log-Likelihood. Es gilt:

$$L = \int q(x_0)\ \log(p_\theta(x_{0:T}))\, dx_0, \tag{2.11}$$

zu maximieren. Diese Maximierung ist jedoch nicht direkt berechenbar [69], weswegen stattdessen der Evidence Lower Bound (ELBO) maximiert wird [30, 69]. Der ELBO stellt dabei eine untere Schranke der Log-Likelihood dar. Durch das Maximieren des ELBO wird somit indirekt die Log-Likelihood maximiert [69]. Somit gilt es, folgenden Term zu maximieren [69]:

$$\begin{aligned} L \leq K = \sum_{t=2}^{T} \iint q(x_0, x_t)\ D_{\mathrm{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \,\Big\|\, p_\theta(x_{t-1} \mid x_t)\Big)\, dx_0\, dx_t \\ + H_q(X_T \mid X_0) - H_q(X_1 \mid X_0) - H_{p_\theta}(X_T), \end{aligned} \tag{2.12}$$

wobei die Terme $H_q(X_T \mid X_0)$ und $H_q(X_1 \mid X_0)$ bedingte Entropien des Vorwärts-

prozesses sind. Die Entropie einer Verteilung P ist definiert als:

$$H(P) = -\int P(x) \log P(x)\, dx, \tag{2.13}$$

und misst die Unsicherheit einer Zufallsvariablen. Im Fall von den beiden Termen $H_q(X_T \mid X_0)$ und $H_q(X_1 \mid X_0)$ wird die Unsicherheit der Zustände X_T beziehungsweise X_1 gegeben dem Startpunkt X_0 gemessen.

Diese Terme sind analytisch berechenbar und bezüglich der Optimierung Konstanten, da der Vorwärtsprozess fixiert ist und nicht von den Modellparametern abhängt. Der Term $H_{p_\theta}(X_T)$ beschreibt die Entropie der Prior-Verteilung des Endzustands des Vorwärtsprozesses und somit auch des Startzustandes des Reverse-Prozesses. Da diese Prior-Verteilung in Diffusionsmodellen üblicherweise als $\mathcal{N}(0, I)$ festgelegt ist, ist die Entropie ebenfalls konstant und unabhängig von den Modellparametern. Diese Terme haben somit keinen Einfluss auf die Optimierung und können für das Training vernachlässigt werden. Da die Kullback-Leibler-Divergenz nichtnegativ ist, entspricht die Maximierung der ELBO der Minimierung der verbleibenden Kullback-Leibler-Divergenz (KLD)-Terme [69]. Das Trainingsziel eines Diffusionsmodells reduziert sich daher auf:

$$\mathcal{L}_{\text{train}}(\theta) = \sum_{t=2}^{T} \mathbb{E}_{q(x_0, x_t)} \left[D_{\text{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \,\Big\|\, p_\theta(x_{t-1} \mid x_t) \Big) \right]. \tag{2.14}$$

Der Term $D_{\text{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \,\Big\|\, p_\theta(x_{t-1} \mid x_t) \Big)$ ist die KLD zwischen der wahren bedingten Verteilung des Reverse-Schritts und der vom Modell approximierten Verteilung. Die KLD misst dabei die Differenz zwischen den beiden Wahrscheinlichkeitsverteilungen. Da es sich bei beiden Verteilungen um gaußsche Verteilung handelt lässt sich die KLD analytisch berechnen. Sie hängt im Wesentlichen von der Differenz der Mittelwerte und Varianzen der beiden Verteilungen ab.

Algorithmus 1 Training eines Denoising Diffusion Probabilistic Model (DDPM) [31]

Initialisiere Modellparameter θ
while nicht konvergiert **do**
  sample $x_0 \sim \mathcal{D}_{\text{train}}$
  sample $t \sim \mathcal{U}(\{1, \ldots, T\})$
  sample $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Gradientenabstieg auf: $\nabla_\theta \left\| \varepsilon - \varepsilon_\theta(\sqrt{\hat{\alpha}_t}\, x_0 + \sqrt{1 - \hat{\alpha}_t}\, \varepsilon, t) \right\|^2$

Obwohl die theoretische Herleitung nahelegt, dass das Diffusionsmodell zwei Funktionsapproximatoren trainiert, die den Mittelwert $\mu_\theta(x_t, t)$ als auch die Varianz $\Sigma_\theta(x_t, t)$ des Reverse-Schritts approximieren, wird das Training in der Praxis anders umgesetzt. Jonathan Ho et al. zeigten in [31], dass unter der Bedingung, dass die Varianz festgelegt wird als:

$$\Sigma_{p_\theta}(x_t, t) = \sigma_t^2 \mathbf{I}, \tag{2.15}$$

wobei σ_t^2 eine vom Zeitschritt t abhängige Konstante ist, die KLD aus Gleichung 2.12, mithilfe der Formel der KLD für Normalverteilungen reduziert werden kann zu:

$$\begin{aligned}\mathcal{D}_t(\theta) &:= D_{\mathrm{KL}}\Big(q(x_{t-1} \mid x_t, x_0) \;\Big\|\; p_\theta(x_{t-1} \mid x_t)\Big) \\ &= \frac{1}{2\sigma_t^2}\|\mu_q(x_t, x_0) - \mu_{p_\theta}(x_t, t)\|^2 + C,\end{aligned} \tag{2.16}$$

wobei C eine Konstante ist, die nicht von den Modellparametern abhängt und $\mu_q(x_t, x_0)$ der Mittelwert der wahren Verteilung des Reverse-Schritts ist. Somit reduzieren sich das Training des Diffusionsmodells darauf, den Mittelwert des Reverse-Schritts korrekt vorherzusagen [31]. Durch weitere Vereinfachungen und Reparametrisierungen des Trainingsziels lässt sich ein anderes Trainingsziel ableiten [31, 71]. Setzt man für x_t, x_t in Abhängigkeit von x_0 und ε ein, wobei ε gegeben ist durch (vgl. Gleichung 2.7):

$$x_t(x_0, \varepsilon) = \sqrt{\hat{\alpha}_t} \cdot x_0 + \sqrt{1 - \hat{\alpha}_t} \cdot \varepsilon, \tag{2.17}$$

in $\mu_q(x_t, x_0)$ ein, erhält man:

$$\mu_q(x_t, x_0) = \mu_q(x_t(x_0, \varepsilon), \frac{1}{\sqrt{\hat{\alpha}_t}}(x_t(x_0, \varepsilon)) - \sqrt{1 - \hat{\alpha}_t} \cdot \varepsilon). \tag{2.18}$$

Da zusätzlich für den echten Mittelwert des Reverse-Schritts gilt, dass er sich aus dem Posterior des Vorwärtsprozesses ableiten lässt, ist er analytisch berechenbar. Dies ist möglich, da der Vorwärtsprozess ein linearer Gaußscher Markov-Prozess ist. Der Mittelwert hängt dabei nur von den bekannten Größen x_0 und x_t sowie den festgelegten Rauschparametern ab. Es ergibt sich eine gewichtete Kombination dieser beiden Zustände [31]:

$$\mu_q(x_t, x_0) = \frac{\sqrt{\hat{\alpha}_{t-1}}\beta_t}{1 - \hat{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1 - \hat{\alpha}_{t-1})}{1 - \hat{\alpha}_t} x_t. \tag{2.19}$$

Setzen wir nun Gleichung 2.19 und Gleichung 2.18 in die Gleichung der KLD ein, erhalten wir [31]:

$$\mathcal{D}_t(\theta) = \frac{1}{2\sigma_t^2}\|\frac{1}{\sqrt{\alpha_t}}(x_t(x_0, \varepsilon) - \frac{\beta_t}{\sqrt{1 - \hat{\alpha}_t}}\varepsilon) - \mu_{p_\theta}(x_t(x_0, \varepsilon), t)\|^2 + C. \tag{2.20}$$

Dabei gibt die Gleichung an, dass der Funktionsapproximator für den Mittelwert $\frac{1}{\sqrt{\alpha_t}}(x_t(x_0, \varepsilon) - \frac{\beta_t}{\sqrt{1-\hat{\alpha}_t}}\varepsilon)$ vorhersagen muss. Da x_t bekannt ist, und dem Modell als Input zur Verfügung steht, lässt sich eine alternative Parametrisierung definieren:

$$\mu_{q_\theta}(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1 - \hat{\alpha}_t}}\varepsilon_\theta(x_t, t)). \tag{2.21}$$

Setzen wir diese alternative Parametrisierung in Gleichung 2.20 ein, erhalten wir [31]:

$$\mathcal{D}_t(\theta) = \frac{\beta_t^2}{2\sigma_t^2\alpha_t(1 - \hat{\alpha}_t)}\|\varepsilon - \varepsilon_\theta(\sqrt{\hat{\alpha}_t}x_0 + \sqrt{1 - \hat{\alpha}_t}\varepsilon, t)\|^2 + C. \tag{2.22}$$

Dabei gibt die Gleichung an, dass der Funktionsapproximator $\varepsilon_\theta(x_t, t)$ das effektive Gesamtrauschen ε, das über alle Zeitschritte bis zum Zeitschritt t hinzugefügt wurde, vorhersagt [31]. Dieses Gesamtrauschen bestimmt die Abweichung von x_t zum ursprünglichen Datenpunkt x_0 und besitzt im Gegensatz zum Rauschen aus dem letzten Zeitschritt eine konsistente, datenabhängige Struktur. Der Schritt von x_{t-1} nach x_t entsteht im Vorwärtsprozess durch Hinzufügen von zufälligem Rauschen und weist daher keine Struktur auf, die ein Modell verlässlich lernen könnte (vgl. Abbildung 2.3). Wird die Gleichung 2.22 explizit als Trainingsziel formuliert, erhält man [31]:

$$\mathcal{L}_{\varepsilon-\mathrm{pred}}(\theta) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{x_0,\varepsilon}\Big[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\hat{\alpha}_t)}\big\|\varepsilon - \varepsilon_\theta(x_t,t)\big\|^2\Big], \tag{2.23}$$

wobei diese Formulierung als ε-Prediction bezeichnet wird [31, 71]. Die Loss-Funktion entspricht einem gewichteten Mean Squared Error (MSE) zwischen dem vom Modell vorhergesagten Rauschen $\varepsilon_\theta(x_t, t)$ und dem tatsächlichen Rauschterm ε, der im Vorwärtsprozess hinzugefügt wurde. Die Gewichtung hängt von den Rauschparametern sowie dem jeweiligen Zeitschritt t ab [31]. Diese Gewichtung dient dazu, die Beiträge der verschiedenen Zeitschritte an der Loss-Funktion auszugleichen. Ohne Gewichtung könnten Zeitschritte mit besonders großem Rauschen tendenziell größere Fehler verursachen, während Zeitschritte mit geringem Rauschen kleinere Fehler verursachen. Durch die Gewichtung wird sichergestellt, dass das Modell für alle Zeitschritte lernt, das Rauschen zuverlässig vorherzusagen [31]. Die Umsetzung des Losses in der Praxis erfolgt jedoch nicht wie in Gleichung 2.23 dargestellt, durch Aufsummieren des Fehlers über alle Zeitschritte t hinweg. Stattdessen wird Monte-Carlo-Sampling verwendet, um den Erwartungswert zu approximieren. Das bedeutet, dass in jeder Update-Iteration ein zufälliges Trainingsbeispiel gezogen wird, für das nur ein Zeitschritt t zufällig ausgewählt wird. Für das ausgewählte Trainingsbeispiel und den Zeitschritt t wird der MSE berechnet [31]. Das führt zu einer deutlich effizienteren Trainingsiteration, bei der das Modell über viele Iterationen hinweg lernt das Rauschen für alle Zeitschritte t korrekt vorherzusagen. Zusätzlich wird in der Praxis häufig auf die explizite Zeitschritt-Gewichtung verzichtet, da das zufällige Sampling einzelner Zeitschritte über viele Iterationen hinweg dazu führt, dass das Modell dennoch lernt, das Rauschen für alle Zeitschritte zuverlässig vorherzusagen [31]. Dadurch ergibt sich der praktische Trainingsalgorithmus für die ε-Prediction wie in Algorithmus 1 dargestellt [31]. Neben der ε-Prediction gibt es auch alternative Parametrisierungen des Trainingsziels. Sie sind mathematisch äquivalent, unterscheiden sich jedoch in Bezug auf ihre Gewichtung und Anforderungen an den Funktionsapproximator [71, 75].

Eine alternative Parametrisierung des Trainingsziels ist die sogenannte x_0-Prediction [31, 71, 75]. Das Trainingsziel wird so formuliert, dass der Funktionsapproximator $x_{0_\theta}(x_t, t)$ den ursprünglichen, nicht verrauschten Datenpunkt x_0 direkt aus dem verrauschten Zustand x_t vorhersagt. Die Umformulierung ist möglich, da der Zustand x_t im Vorwärtsprozess eine lineare Funktion des

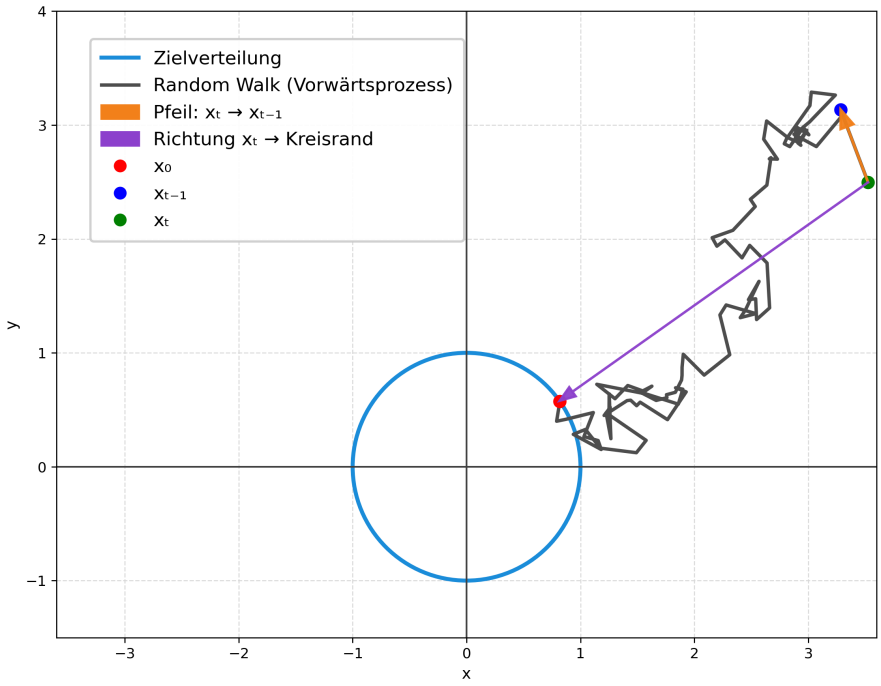

Abbildung 2.3.: Visualisierung eines zweidimensionalen Random-Walk-Vorwärtsprozesses (schwarz) relativ zu einer Zielverteilung am Kreisrand (blau). Dargestellt sind der aktuelle Zustand x_t (grün), der vorherige Zustand x_{t-1} (blau) mit dem Übergang $x_t \to x_{t-1}$ (orange) sowie die geometrische Richtung von x_t zum Zielpunkt x_0 am Kreisrand (violett, x_0 rot) (inspiriert durch [76]).

ursprünglichen Datenpunkts x_0 und des hinzugefügten Gaußschen Rauschens ε ist (vgl. Gleichung 2.17). Durch Umstellen dieser Gleichung lässt sich das Rauschen ε explizit als Funktion von x_t und x_0 darstellen:

$$\varepsilon(x_t, x_0) = \frac{1}{\sqrt{1-\hat{\alpha}_t}} \left(x_t - \sqrt{\hat{\alpha}_t}\, x_0(x_t, \varepsilon) \right). \tag{2.24}$$

Setzt man diese Darstellung für ε in das ε-Prediction Trainingsziel aus Gleichung 2.23 ein und ersetzt den unbekannten Term x_0 durch die vom Modell vorhergesagte Größe $x_{0,\theta}(x_t, t)$, ergibt sich ein äquivalentes Trainingsziel. Bei diesem Trainingsziel rekonstruiert das Modell den ursprünglichen Datenpunkt direkt. Das resultierende Trainingsziel der x_0-Prediction lautet:

$$\mathcal{L}_{x_0\text{-pred}}(\theta) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{x_0,\varepsilon} \left[\frac{\hat{\alpha}_t}{1-\hat{\alpha}_t} \left\| x_0 - x_{0_\theta}(x_t, t) \right\|^2 \right]. \tag{2.25}$$

Auch hier wird in der Praxis für jedes Trainingsbeispiel ein zufälliger Zeitschritt t ausgewählt und für das vorhersagte x_0 und das echte x_0 der MSE zur Modelloptimierung berechnet [71, 75].

Neben der ε-Prediction und der x_0-Prediction existiert auch die sogenannten v-Prediction [71, 77]. Die v-Prediction ist dabei eine weitere, mathematisch äquivalente Parametrisierung des Trainingsziels, die einer linearen Kombination von ε und x_0 entspricht [77]. Das Trainingsziel der v-Prediction lautet dabei [71, 75]:

$$\mathcal{L}_{v\text{-pred}}(\theta) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{x_0,\varepsilon} \left[\left\| v - v_\theta(x_t, t) \right\|^2 \right], \tag{2.26}$$

und v definiert ist als [71, 75]:

$$v = \sqrt{\hat{\alpha}_t}\varepsilon - \sqrt{1-\hat{\alpha}_t} x_0, \tag{2.27}$$

wobei $v_\theta(x_t, t)$ der Funktionsapproximator ist, der v aus dem verrauschten Zustand x_t vorhersagt [71, 75].

Aus der Herleitung der drei in Diffusionsmodellen verwendeten Trainingsziele folgt, dass sie mathematisch äquivalent sind und denselben Rekonstruktionsfehler minimieren. Sie unterscheiden sich nur in ihrer Gewichtung sowie in den Anforderungen an den Funktionsapproximator. Durch diese Äquivalenz ist es möglich, ein Modell so zu trainieren, dass es eine Vorhersage in einem bestimmten Ausgaberaum, beispielsweise im ε-Raum, erzeugt, während der Optimierungsprozess formal den Fehler in einem anderen Raum, etwa dem x_0-Raum, minimiert. Diese Eigenschaft ergibt sich daraus, dass sich die drei Zielgrößen ineinander umrechnen lassen und sich somit lediglich die Gewichtung der Fehlerterme unterscheidet [71, 75]. Aus dieser Äquivalenz könnte man schließen, dass die Wahl des Trainingsziels keinen Einfluss auf die Qualität der generierten Daten hat. In der Praxis gilt das jedoch nicht. Der Grund dafür liegt darin, dass sich die Struktur

der Zielräume unterscheidet und damit auch die Art der Funktion, die vom Modell gelernt werden muss. Bei der ε-Prediction sagt das Modell Rauschen voraus, das unabhängig und normalverteilt ist. Diese Eigenschaft erleichtert sowohl die Analyse als auch die Stabilität des Trainings, da ein gut trainierter ε-Predictor Ausgaben liefern sollte, die einer Normalverteilung folgen. Im Gegensatz dazu besitzt der x_0-Raum eine deutlich komplexere, jedoch strukturierte Struktur. Sie kommt daher, dass der x_0-Raum die ursprünglichen Daten repräsentiert und somit lokale Abhängigkeiten und semantische Zusammenhänge enthalten kann [75]. Diese Eigenschaft steht in engem Zusammenhang mit der Mannigfaltigkeitsannahme [78, 79]. Die Mannigfaltigkeitsannahme besagt, dass hochdimensionale natürliche Daten, wie Bilder, Audiosignale oder Texte, den möglichen Datenraum nicht vollständig ausfüllen, sondern sich auf einer niedrigdimensionalen Mannigfaltigkeit konzentrieren. In den hochdimensionalen Räumen existieren daher viele Bereiche, die keine gültigen Datenpunkte enthalten. Diese Struktur ergibt sich dadurch, dass natürliche Daten bestimmten Regeln folgen [78–80]. Ein Beispiel findet sich im Bereich der Sprache. Hier wird der hochdimensionale Raum aller möglichen Wortfolgen nicht komplett ausgefüllt da natürliche Sätze überwiegend auf einer niedrigdimensionalen Mannigfaltigkeit grammatikalisch und semantisch sinnvoller Sätze liegen [80]. Für die x_0-Prediction bedeutet dies, dass der Funktionsapproximator die zugrundeliegende niedrigdimensionale Struktur der Daten lernen muss, um den ursprünglichen Datenpunkt korrekt zu rekonstruieren. Die Struktur des x_0-Raums kann dazu führen, dass die Wahl des Trainingsziels einen spürbaren Einfluss auf das Modellverhalten hat. Insbesondere für Modelle mit begrenzter Kapazität kann es einfacher sein, eine strukturierte, niedrigdimensionale Mannigfaltigkeit zu lernen als unstrukturiertes Rauschen zu approximieren [75]. Diese Beobachtung wird durch die Ergebnisse von Li et al. [75] gestützt. Die Autoren zeigen experimentell, dass die Wahl des Trainingsziels einen signifikanten Einfluss auf die Qualität der generierten Daten hat. Dieser Einfluss tritt vor allem auf, wenn die Modellkapazität im Verhältnis zur Komplexität der Datenverteilung begrenzt ist. In diesen Fällen führt das Training mit x_0-Prediction zu einer deutlich besseren Generierungsqualität als bei der Verwendung von ε-Prediction oder v-Prediction.

2.3.1.2. Sampling

Obwohl die unterschiedlichen Trainingsziele darauf ausgerichtet sind, den Rekonstruktionsfehler zu minimieren, besteht das eigentliche Ziel eines generativen Modells nicht in der Wiederherstellung eines gegebenen Datenpunkts. Stattdessen soll das Diffusionsmodell neue Stichproben erzeugen, die der zugrundeliegenden Datenverteilung entsprechen. Im Trainingsprozess lernt das Modell, für jeden Zeitschritt t die Umkehrung des Vorwärtsprozesses zu approximieren. Dadurch werden die verrauschten Startzustände sukzessive in Zustände mit hoher Wahrscheinlichkeit transformiert [31, 32, 69].

Für die Generierung neuer Datenpunkte existieren verschiedene Sampling-Methoden, die sich unter anderem in Geschwindigkeit, Deterministik, Stabilität

Algorithmus 2 Sampling mit DDPM [31]

sample $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, T-1, \ldots, 1$ **do**
  sample $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ **if** $t > 1$, **else** $z \leftarrow 0$
  $x_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\hat{\alpha}_t}}\,\varepsilon_\theta(x_t, t)\right) + \sigma_t z$
return x_0

und Qualität unterscheiden [31, 32, 81]. Die grundlegende Methode ist das von Ho et al. [31] eingeführte DDPM-Sampling. Um mithilfe von DDPM neue Datenpunkte zu erzeugen, wird der im Training gelernte Reverse-Diffusionsprozess angewandt. Während des Trainings lernt das Modell, ein verrauschter Zustand x_t mithilfe des Funktionsapproximators in einen strukturierten Datenpunkt oder eine äquivalente Repräsentation zu transformieren [31, 75]. Dabei ist der Vorwärtsprozess so konstruiert, dass der Endzustand x_T und somit auch der Startzustand des Reverse-Prozesses einer Standardnormalverteilung $\mathcal{N}(0, I)$ folgt. Dies ermöglicht es, den Sampling-Prozess durch Ziehen einer Stichprobe $x_T \sim \mathcal{N}(0, I)$ zu initialisieren. Anschließend wird der gelernte Reverse-Diffusionsprozess schrittweise angewandt. Für jeden Zeitschritt t bestimmt das Modell aus dem aktuellen Zustand x_t und dem Zeitindex t eine Vorhersage des ursprünglichen Datenpunkts x_0 (bei x_0-Prediction) oder des im Vorwärtsprozess hinzugefügten Gesamtrauschens ε (bei ε-Prediction). Beide Größen sind über Gleichung 2.17 ineinander umrechenbar [31, 75]. Die Vorhersage wird anschließend genutzt, um mit Gleichung 2.19 den Mittelwert des Reverse-Schritts zu berechnen. Da die Varianz des Reverse-Schritts während des Trainings festgelegt wurde, ist auch sie für jeden Zeitschritt t bekannt. Dadurch lässt sich eine Stichprobe aus der approximierten bedingten Verteilung $p_\theta(x_{t-1}|x_t, t)$ ziehen. Es gilt:

$$x_{t-1} = \mu_{p_\theta}(x_t, t) + \Sigma_{p_\theta}(x_t, t)^{\frac{1}{2}} \cdot z, \tag{2.28}$$

wobei $z \sim \mathcal{N}(0, I)$ eine Stichprobe aus der Standardnormalverteilung ist, $\mu_{p_\theta}(x_t, t)$ der vom Modell vorhergesagte Mittelwert und $\Sigma_{p_\theta}(x_t, t)$ die festgelegte Varianz des Reverse-Schritts ist [31]. Dieser Vorgang wird für alle Zeitschritte $t = T, T-1, \ldots, 1$ wiederholt, um schrittweise von der initialen Stichprobe x_T zu einem neuen Datenpunkt x_0 zu gelangen. Somit ist der DDPM-Sampling-Algorithmus beispielhaft mit einem ε Prediction Model wie folgt definiert:

Der zufällig normalverteilte Term z wird in jedem Zeitschritt hinzugefügt, da der Vorwärtsprozess und somit auch der Reverse-Schritt stochastisch sind. Daraus folgt, dass der komplette DDPM-Sampling-Prozess stochastisch ist. Das bedeutet, dass selbst bei identischen Modellparametern und identischem Startwert x_T unterschiedliche Stichproben x_0 generiert werden können [31]. Die diskrete Abbildung des Reverse-Prozesses als stochastischer Markov-Prozess, bei dem alle Zeitschritte sequenziell durchlaufen werden, ist zwar theoretisch fundiert, ist in der Praxis jedoch aufgrund der großen Anzahl benötigter Sampling-Schritte langsam [32].

Um die Sampling-Geschwindigkeit zu erhöhen, führten Song et al. [32] Denoising Diffusion Implicit Model (DDIM) ein. Die Autoren zeigten, dass das Trainingsziel von DDPM ausschließlich von den marginalen Verteilungen $q(x_t \mid x_0)$ abhängt und unabhängig von der vollständigen Verteilung $q(x_{1:T} \mid x_0)$ ist. Auf dieser Grundlage lässt sich ein alternativer Vorwärtsprozess definieren, der dieselben marginalen Verteilungen wie der ursprüngliche Vorwärtsprozess aufweist, dessen gemeinsame Verteilung jedoch anders strukturiert ist und keine Markov-Eigenschaft besitzt [32]. Zu diesem Zweck definieren Song et al. eine Familie gemeinsamer Verteilungen:

$$q_\sigma(x_{1:T} \mid x_0) = q_\sigma(x_T \mid x_0) \prod_{t=2}^{T} q_\sigma(x_{t-1} \mid x_t, x_0), \tag{2.29}$$

mit

$$q_\sigma(x_T \mid x_0) = \mathcal{N}(\sqrt{\alpha_T} x_0, (1-\alpha_T)\mathbf{I}), \tag{2.30}$$

$$q_\sigma(x_{t-1} \mid x_t, x_0) = \mathcal{N}\Big(\sqrt{\alpha_{t-1}} x_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\frac{x_t - \sqrt{\alpha_t} x_0}{\sqrt{1-\alpha_t}},\ \sigma_t^2 \mathbf{I}\Big), \tag{2.31}$$

wobei die Parameter $\sigma_t^2 \in [0, 1-\alpha_{t-1}]$ die Zufälligkeit des Prozesses steuern. Für $\sigma_t^2 = 0$ ergibt sich der deterministische DDIM-Prozess, während die Wahl $\sigma_t^2 = \frac{1-\alpha_{t-1}}{1-\alpha_t}\big(1-\alpha_t\big)$ den ursprünglichen stochastischen, markovschen DDPM-Vorwärtsprozess reproduziert [32]. Die Konstruktion des Prozesses stellt sicher, dass für alle t analog zu DDPM (vgl. Gleichung 2.7) gilt:

$$q_\sigma(x_t \mid x_0) = \mathcal{N}(\sqrt{\alpha_t} x_0, (1-\alpha_t)\mathbf{I}). \tag{2.32}$$

Song et al. zeigen, dass das resultierende Trainingsziel bis auf eine konstante, die nicht von den Modellparametern abhängt, dem DDPM-Trainingsziel entspricht. Sie stellen außerdem fest, dass die Anzahl der verwendeten Diffusionsschritte keinen Einfluss auf das Trainingsziel hat. Damit kann ein mit dem DDPM-Trainingsziel trainiertes Modell unverändert als DDIM-Modell mit einer beliebigen Anzahl an Vorwärtsschritten verwendet werden [32]. Der daraus resultierende Vorteil wird im Sampling-Prozess deutlich. Für den deterministischen Fall $\sigma_t^2 = 0$ gilt, dass x_{t-1} vollständig durch x_t und die Modellvorhersage für x_0 bestimmt werden kann. Der generative Prozess ergibt sich aus [32]:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{x_t - \sqrt{1-\alpha_t}\, \varepsilon_\theta(x_t, t)}{\sqrt{\alpha_t}} \right) + \sqrt{1-\alpha_{t-1}}\, \varepsilon_\theta(x_t, t), \tag{2.33}$$

wobei $\varepsilon_\theta(x_t, t)$ das vom Modell vorhergesagte Gesamtrauschen ist. Obwohl der Reverse-Prozess auf denselben Zeitschritten wie der Vorwärtsprozess basiert, ist das trainierte Modell äquivalent zu einem DDIM-Modell mit variabler Schrittanzahl. Daher kann die Anzahl der Schritte im generativen Prozess frei gewählt werden. Dadurch ist es möglich, deutlich weniger Sampling-Schritte als bei DDPM zu machen und dadurch die Sampling-Geschwindigkeit zu erhöhen. Dabei gilt, dass die Qualität der generierten Daten mit abnehmender Anzahl an Sampling-Schritten

sinkt und sich ein Kompromiss zwischen Qualität und Geschwindigkeit ergibt [32]. Auch wenn durch DDIM die Sampling-Geschwindigkeit im Vergleich zu DDPM bei gleichbleibender Qualität um den Faktor 10 bis 50 erhöht werden kann [32], bleibt die Generierung im Vergleich zu anderen generativen Modellen langsam. Diffusionsmodelle erzeugen Daten schrittweise, indem Rauschen in mehreren aufeinanderfolgenden Schritten entfernt wird. Selbst bei DDIM kann dieser Prozess nicht auf einen einzelnen Schritt verkürzt werden, ohne die Qualität der generierten Daten deutlich zu beeinträchtigen. Dies liegt daran, dass die Modelle eine stochastische oder deterministische Denoising-Dynamik approximieren, deren Score-Funktion nur lokale Informationen über die Wahrscheinlichkeitsdichte liefert. Um komplexe, hochdimensionale Datenverteilungen zu approximieren, ist eine schrittweise Anwendung der Denoising Schritte erforderlich. Modelle wie GANs oder VAEs lernen dagegen direkte Abbildungen zwischen einem latenten Raum und dem Datenraum. Dadurch können sie mit einem einzigen Schritt neue Datenpunkte generieren [29, 30]. Aufgrund dieser unterschiedlichen Struktur bleiben Diffusionsmodelle selbst mit beschleunigten Sampling-Verfahren wie DDIM im Vergleich zu anderen generativen Modellfamilien langsam.

2.3.1.3. U-Net als Funktionsapproximator

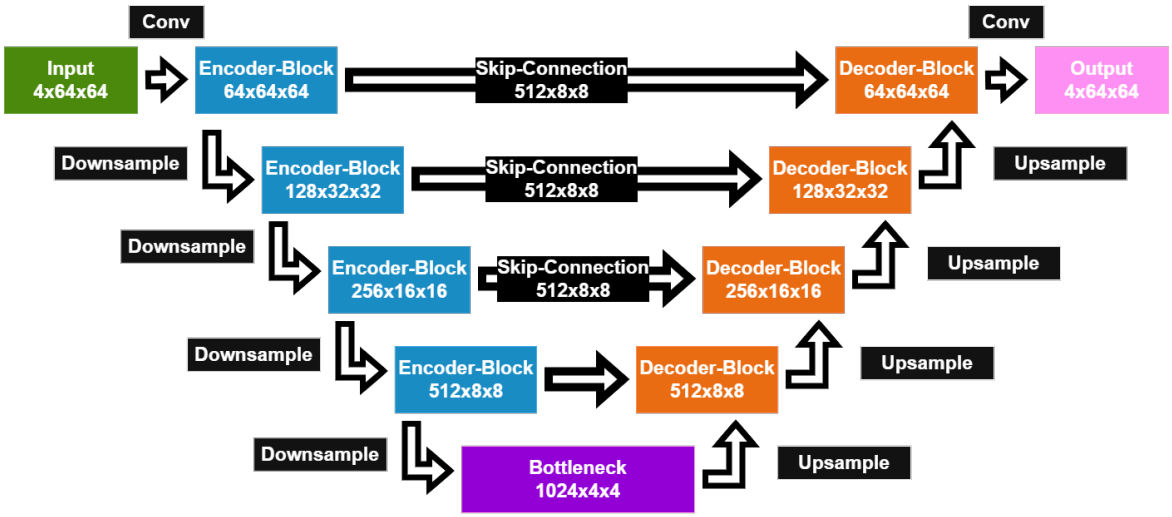

Abbildung 2.4.: Darstellung der U-Net-Architektur, wie sie in Diffusionsmodellen verwendet wird. Die Architektur besteht aus einem Encoder-Teil (links) und einem Decoder-Teil (rechts), die durch Skip-Verbindungen (Pfeile) verbunden sind [82]. Die Dimensionen der Feature-Maps werden durch die Zahlen in den Blöcken dargestellt. Die erste Zahl gibt die Anzahl der Kanäle an und die zweite und dritte die räumliche Auflösung.

Die Modellarchitektur des Funktionsapproximators bildet die lernbare Komponente des Diffusionsmodells. Der Funktionsapproximator lernt abhängig vom Trainingsziel entweder das Gesamtrauschen ε, den ursprünglichen Datenpunkt x_0 oder die

lineare Kombination v aus x_0 und ε vorherzusagen (siehe Abschnitt 2.3.1.1). Die Wahl der Architektur ist dabei nicht explizit vorgegeben. Es bieten sich Transformer-basierte Architekturen [75, 83] sowie klassische Convolutional Neural Network (CNN)-Architekturen an [31, 32, 84, 85]. Die Wahl der Architektur beeinflusst die Leistungsfähigkeit des Diffusionsmodells, da sie die Kapazität und die Ausdrucksfähigkeit des Modells bestimmen [31].

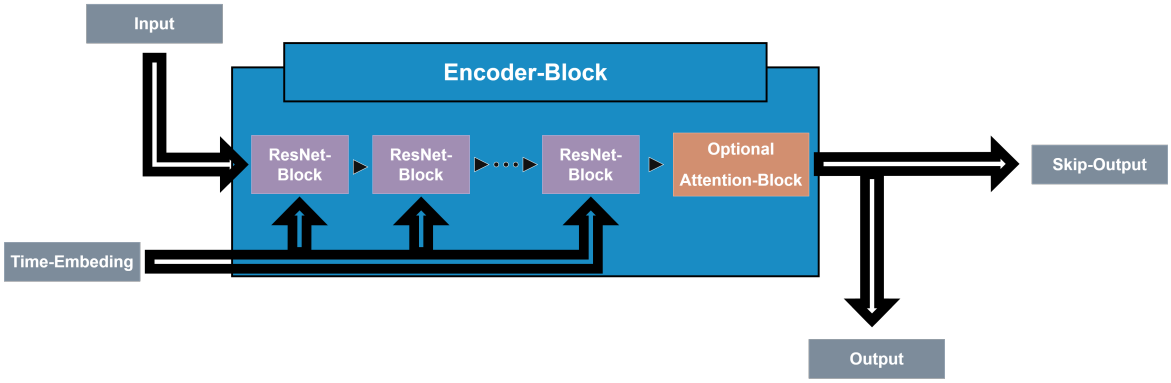

Abbildung 2.5.: Interne Struktur des Encoder-Blocks, die aus einer beliebigen Anzahl von ResNet-Blöcken besteht und optional durch einen Self-Attention-Block ergänzt wird.

Unter den möglichen Architekturen nimmt das U-Net eine besondere Rolle ein, da es sich in zahlreichen Anwendungen als effektiv erwiesen hat [31, 84, 85]. Ein U-Net ist eine CNN-Architektur, die ursprünglich für die Bildsegmentierung entwickelt wurde [82]. Sie zeichnet sich durch ihre symmetrische Struktur aus, die wie ein U dargestellt werden kann (siehe Abbildung 2.4). Das U-Net besteht dabei aus einem Encoder- und einem Decoder-Teil, die durch Skip-Verbindungen miteinander verbunden sind. Der Encoder und der Decoder bestehen jeweils aus mehreren Blöcken. Die Blöcke im Encoder sind dabei mit Downsampling-Operationen verbunden, die die räumliche Auflösung der Eingabe schrittweise reduzieren und dabei die Anzahl der Kanäle erhöhen. Das Downsampling erfolgt dabei üblicherweise durch eine Convolution-Operation mit einer Schrittweite von 2, welche die räumlichen Dimensionen halbiert. Gleichzeitig wird die Anzahl der Kanäle erhöht, indem doppelt so viele Filter wie in der vorherigen Ebene vorhandene Kanäle verwendet werden [82]. Decoder-Blöcke sind mit Upsampling-Operationen verbunden. Sie stellen die räumliche Auflösung wieder her und reduzieren die Anzahl der Kanäle. Das Upsampling erfolgt dabei durch eine zweistufige Transformation. Als erstes wird die räumliche Auflösung der Feature-Maps vergrößert, indem zum Beispiel Nearest-Neighbor-Upsampling durchgeführt wird. Beim Nearest-Neighbor-Upsampling werden neu entstehende Pixel durch den Wert des jeweils nächstgelegenen Originalpixels ersetzt. Die zweite Transformation ist eine Convolution-Operation, die die Kanalanzahl durch das Verwenden von halb so vielen Filtern wie der Eingabe halbiert [82]. Dadurch ergeben sich im U-Net verschiedene Ebenen gleicher räumlicher Auflösung und Kanalanzahl, die durch die Skip-Verbindungen direkt miteinander verbunden werden. Durch

diese Verbindungen kann das Modell sowohl lokale als auch globale Informationen berücksichtigen [82].

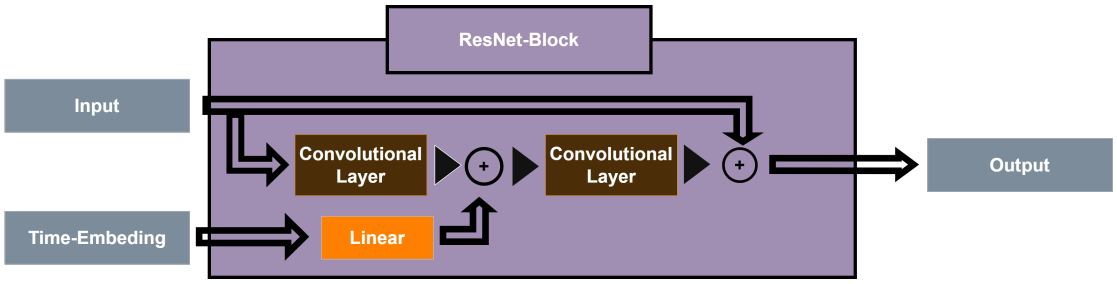

Abbildung 2.6.: Interne Struktur eines ResNet-Blocks wie er in U-Net-Diffusionsmodellen verwendet wird. Die Darstellung lässt Normalisierungs- und Aktivierungsoperationen zugunsten der Übersichtlichkeit bewusst aus.

Die Architektur der verwendeten Encoder-Blöcke, Decoder-Blöcke und der tiefsten Schicht unterscheidet sich dabei von der ursprünglichen U-Net-Struktur nach Ronneberger et al. [82]. In Diffusionsmodellen werden diese Blöcke erweitert. Anstelle von einfacher Convolution-Schichten werden komplexere ResNet-Blöcke hintereinandergeschaltet [31, 84]. ResNet-Blöcke besitzen gegenüber herkömmlichen sequenziellen Convolution-Schichten den Vorteil, dass sie eine Skip-Connection enthalten, welche den ursprünglichen Eingang direkt mit dem Ausgang des Blocks verbindet. Durch diese Struktur muss der Block nicht mehr die vollständige Transformation des Eingangssignals lernen, sondern lediglich die Abweichung vom Eingang und Output. Die Skip-Connection erleichtert das Lernen erheblich, da Identitätsabbildungen und kleine Funktionsänderungen einfacher optimiert werden können. Darüber hinaus verbessert die Skip-Connection den Gradientenfluss, da ein Teil des Gradienten unverändert über die Skip-Connection zurückpropagiert werden kann. Dadurch können die Modellparameter auch in sehr tiefen Schichten optimiert werden, ohne dass es zum Vanishing-Gradient-Problem kommt [86].

Zusätzlich wird in modernen U-Nets in einigen, insbesondere den tiefer liegenden Encoder- und Decoder-Blöcken sowie dem Bottleneck, Self-Attention eingesetzt [31, 84]. Self-Attention ist ein Mechanismus, der es jedem Element der Eingabe ermöglicht, seinen Wert durch eine gewichtete Summe aller anderen Elemente zu aktualisieren. Die Gewichte stammen aus einem berechneten Relevanzmaß zwischen den Elementen. Die Umsetzung dieses Mechanismus erfolgt, indem jedes Eingabeelement über drei Transformationen in Query-, Key- und Value-Vektoren abgebildet wird [57]. Der Query-Vektor eines Elements dient dabei als aufmerksamkeitslenkende Größe. Im Self-Attention-Mechanismus wird er mit den Key-Vektoren aller anderen Elemente der Sequenz über ein skaliertes Skalarprodukt verknüpft. Die daraus resultierenden Ähnlichkeitswerte werden anschließend durch eine Softmax Funktion in eine normalisierte Gewichtsmatrix überführt. Sie gibt für jedes Element an, wie relevant alle anderen Elemente für dessen Aktualisierung sind. Die Gewichtungsmatrix wird anschließend mit den Value-Vektoren multipliziert,

wodurch eine Ausgabematrix entsteht, deren Zeilen die relevanzgewichteten Value-Vektoren repräsentieren. Formal ergibt sich nach [57]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V. \tag{2.34}$$

Da im Rahmen von Diffusionsmodellen mit Self-Attention auf dreidimensionalen Tensoren $X \in \mathbb{R}^{C \times H \times W}$ gearbeitet wird, werden diese zunächst durch ein Flattening der räumlichen Dimensionen in eine Sequenz von $H \cdot W$ Positionen überführt. Danach kann der Self-Attention-Mechanismus wie beschrieben angewendet werden [87]. In Diffusionsmodellen ergänzen Self-Attention-Schichten die Encoder- und Decoder-Blöcke um die Modellierung globaler räumlicher Abhängigkeiten [87].

Zusätzlich bekommen die Encoder- und Decoder-Blöcke in U-Net-Diffusionsmodellen den aktuellen Zeitschritt t als Eingabe [31]. Dafür wird der aktuelle Zeitschritt t zunächst mit einem sinusförmigen Positional-Encoding in einen hochdimensionalen Vektor überführt. Die Komponenten des Vektors bestehen aus Sinusfunktionen und Cosinusfunktionen verschiedener Frequenzen. Diese Kodierung gewährleistet, dass kleine Änderungen im Zeitschritt t in der Praxis zu lokal stetigen Veränderungen des Embeddings führen [31, 57]. Das resultierende Zeit Embedding wird in jeden Encoder- und Decoder-Block hineingegeben. Dadurch erhält das Modell Informationen über die aktuelle Rauschstufe des Diffusionsprozesses. Die Eingabe erfolgt in dem das Zeit Embedding in jeden ResNet-Block additiv hinzugefügt wird (vgl. Abbildung 2.6). Insgesamt ergibt sich die Architektur der Decoder- beziehungsweise Encoder-Blöcke wie in Abbildung 2.5 dargestellt.

2.3.1.4. Tricks of the Trade

Ein zentraler Faktor, der über die Struktur des Diffusionsmodells hinausgeht, ist die Wahl der Noise-Schedule. Sie bestimmt, wie viel Rauschen in jedem Forward-Schritt hinzugefügt wird. Dabei muss sichergestellt werden, dass nach T Schritten die ursprüngliche Datenstruktur praktisch vollständig zerstört ist und nur noch standardnormalverteiltes Rauschen verbleibt. Das bedeutet, dass die kumulative Rauschmenge in Bezug auf den kompletten Diffusionsprozess groß genug sein muss. Gleichzeitig darf die Information in den frühen Schritten nicht zu schnell verloren gehen, damit das Modell feine Details präzise rekonstruieren kann [84]. Ein erster Ansatz ist eine lineare Noise-Schedule. Sie beginnt mit kleinem Rauschen im ersten Schritt $t = 1$ und erhöht das hinzugefügte Rauschen in jedem darauffolgenden Schritt linear [31]. Dadurch bleibt in den ersten Schritten viel Information erhalten, während in den späteren Schritten zunehmend stärkeres Rauschen hinzugefügt wird, um schlussendlich einen komplett verrauschten Zustand für $t = T$ zu erhalten. Das Verhalten der linearen Noise-Schedule ist in Abbildung 2.7 in Bezug auf den kumulativen Signalanteil visualisiert. Hier erkennt man, dass das Signal unter einer linearen Noise-Schedule nicht gleichmäßig, sondern nichtlinear abnimmt. Die Informationen bleiben in frühen Diffusionsschritten

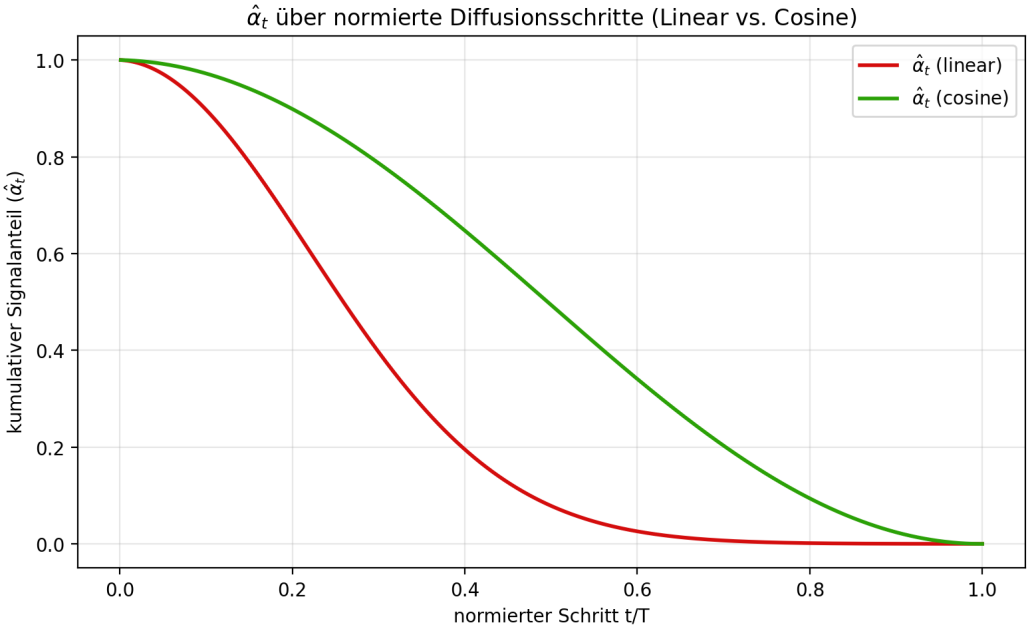

Abbildung 2.7.: Die Abbildung zeigt den noch verbleibenden Anteil des ursprünglichen Signals $\bar{\alpha}_t$ über normierte Diffusionsschritte t/T für die Lineare und Cosine-Noise-Schedule.

weitgehend erhalten, während der Signalanteil in späteren Schritten vergleichsweise schnell zerstört wird. Abbildung 2.8 zeigt dieses Verhalten in Bezug auf Bilder. Aus dem Grund des nichtlinearen Informationsverlust, der durch das Verwenden der linearen Noise-Schedule entsteht, führten Nichol und Dhariwal in [84] die Cosine-Noise-Schedule ein. Sie definieren dafür die Funktion:

$$\hat{\alpha}_t = \frac{f(t)}{f(0)}, \qquad f(t) = \cos\left(\frac{\frac{t}{T} + s}{1+s} \cdot \frac{\pi}{2}\right)^2, \tag{2.35}$$

wobei s der Offset-Parameter, der dafür sorgt, dass das hinzugefügte Rauschen im ersten Schritt nicht zu klein ist. Die $\cos^2$ Funktion ist eine geeignete Funktion, da sie zu einem über weite Bereiche näherungsweise linearen Abfall des noch verbleibenden Anteils des ursprünglichen Signals über alle Zeitschritte hinweg führt. Die lineare Abnahme wird nur im Bereich der ersten und der letzten Zeitschritte abgeflacht. Abbildung 2.7 verdeutlicht dieses Verhalten. Frühe Diffusionsschritte halten mehr Information zurück, sodass das Modell feine Details aus gering verrauschten Daten lernen kann, während späte Schritte das Signal zuverlässig in Richtung nahezu reinen Rauschens führen. Damit verteilt die Cosine-Noise-Schedule den Informationsverlust über die gesamte Zeit, gleichmäßiger als eine lineare Noise-Schedule, erreicht gegen Ende des Vorwärtsprozesses trotzdem einen Zustand, der nur aus Rauschen besteht. Der maximale Verlust pro Schritt wird somit reduziert. In der Praxis führt dies zu stabilerem Training und höherer Qualität der generierten Bilder.

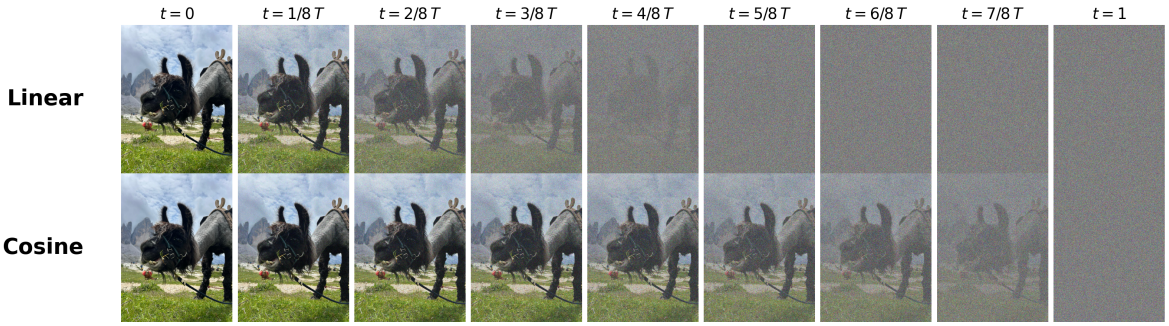

Abbildung 2.8.: Die Abbildung zeigt ein Bild, das mithilfe der linearen Noise-Schedule (oben) und der Cosine-Noise-Schedule (unten) verrauscht wurde [84].

Ein weiterer Trick, der nachweislich zu einer höheren Qualität der generierten Daten beiträgt, ist das Sampling mit einer Exponential Moving Average (EMA) geglätteten Kopie der Modellparameter [31, 84, 88]. Dazu werden während des Trainings nicht nur die aktuellen, frei lernbaren Parameter des Modells aktualisiert, sondern zusätzlich eine zweite Kopie der Gewichte, die sogenannten EMA-Gewichte. Diese werden bei jedem Optimierungsschritt mittels eines exponentiell gewichteten Mittelwerts aktualisiert. Formal ergibt sich das Update durch

$$\theta_{\mathrm{EMA}}^{(t)} = \alpha\, \theta_{\mathrm{EMA}}^{(t-1)} + (1-\alpha)\, \theta^{(t)}, \tag{2.36}$$

wobei $\theta^{(t)}$ die aktuellen Modellparameter, $\theta_{\mathrm{EMA}}^{(t)}$ die geglätteten Parameter und $\alpha \in [0, 1[$ den Glättungsfaktor bezeichnet. Nach Abschluss des Trainings wird für das Sampling nicht das rohe Modell $\theta^{(t)}$ verwendet, sondern die geglättete Kopie $\theta_{\mathrm{EMA}}^{(t)}$. Die Verwendung von EMA für Modellparameter ist dabei eine Methode, die nicht ausschließlich in Diffusionsmodellen angewendet wird. Vielmehr handelt es sich um ein allgemeines Verfahren zur Stabilisierung des Trainings in neuronalen Netzen, insbesondere in Kombination mit stochastischem Gradientenabstieg und den damit verbundenen rauschbehafteten Parameter-Updates. EMA wirkt dabei als eine Form impliziter Regularisierung und verbessert die Stabilität der Modellgeneralisation [89]. Gerade in Diffusionsmodellen spielt diese Glättung der Parameter eine besondere Rolle, da die Updates in Diffusionsmodellen verrauscht sind. Ein Grund dafür ist die zufällige Auswahl des Zeitschritts t, mit dem die Schwierigkeit des Trainingsziels stark variiert. Ein anderer Grund ist die zufällige Auswahl der Trainingsdaten im jeweiligen Optimierungsschritt. Dadurch repräsentieren die rohen Gewichte $\theta^{(t)}$ oft nur einen verrauschten Momentzustand des Trainingsprozesses, während die EMA-Gewichte einen geglätteten, zeitlich integrierten Schätzer des tatsächlichen Modellverhaltens darstellen. Aus diesem Grund werden in der Praxis die finalen Samples moderner Diffusionsmodelle typischerweise mit den EMA-Gewichten erzeugt [31, 84, 88].

2.4. Evaluierung von synthetischen Daten

Generative Modelle zielen darauf ab, Daten zu erzeugen, deren statistische Verteilung möglichst gut mit der Verteilung der Trainingsdaten übereinstimmt. Zur Bewertung dieser Daten sind jedoch geeignete Evaluationsmethoden erforderlich. Mit diesen Evaluationsmethoden lassen sich zwei Modelle miteinander vergleichen oder die Validität der erzeugten Daten nachweisen. Dafür reicht das Erreichen des Trainingsziels, in Bezug auf einen niedrigen Trainingsfehler des Modells nicht aus. Stattdessen ist eine externe Evaluation notwendig, die mithilfe spezifischer Metriken oder Distanzmaßen erfolgt. Eine Metrik ist ein Distanzmaß, das nichtnegativ ist, nur für identische Elemente den Wert Null annimmt, symmetrisch ist und die Dreiecksungleichung erfüllt [90]. Viele der als Evaluationsmaß genutzten Distanzmaße erfüllen nicht alle Eigenschaften.

Ein zentraler Ansatz zur Bewertung synthetischer Daten besteht darin, die statistische Ähnlichkeit zwischen der Verteilung der echten und der generierten Daten zu messen. Hierbei kommen Abstandsmaße wie die KLD, die Jensen-Shannon-Divergenz oder die FID zum Einsatz [20]. Diese Ähnlichkeitsmaße quantifizieren die Distanz zwischen zwei Wahrscheinlichkeitsverteilungen und ermöglichen eine objektive Einschätzung, wie gut das generative Modell die zugrundeliegende Datenverteilung reproduzieren kann. Die Evaluation erfolgt auf der Verteilungsebene und nicht auf der Ebene einzelner Beispiele, was insbesondere im Kontext von Bilddaten relevant ist, da visuelle Qualität allein kein verlässlicher Indikator für statistische Ähnlichkeit ist.

Soll die Qualität einzelner synthetischer Bilder bewertet werden, können sogenannte Diskriminatoren eingesetzt werden. Dabei handelt es sich um Modelle, die darauf trainiert sind, zwischen echten und generierten Bildern zu unterscheiden [14, 16]. Die Motivation dieser Vorgehensweise liegt darin, dass synthetische Daten, die der Originalverteilung stark ähneln, nur schwer durch eine Trennfunktion von echten Daten unterschieden werden können [91]. Ein Diskriminator, der Schwierigkeiten hat, zwischen echten und synthetischen Bildern zu unterscheiden, liefert ein indirektes Maß für die Qualität der erzeugten Bilder. Diese Bewertungsmethode operiert auf der Ebene einzelner Bilder und ist im Besonderen für visuelle Inspektionen sowie sicherheitskritische Anwendungsbereiche relevant [92]. Obwohl generierte Bilder visuell eine hohe Qualität aufweisen und von Menschen nicht zuverlässig von echten Bildern unterscheidbar sind, lassen sie sich durch trainierte Diskriminatoren mit hoher Genauigkeit identifizieren [16].

2.4.1. Inception Score

Der Inception Score ist ein Maß zur Bewertung der Leistungsfähigkeit generativer Bildmodelle [26]. Die Berechnung des Inception Scores erfordert ein vortrainiertes Klassifikationsmodell, das entweder auf den Trainingslabels des generativen Modells [93] oder auf einem externen Datensatz mit überlappenden Klassen trainiert

wurde. Der Inception Score basiert auf zwei zentralen Annahmen. Erstens wird angenommen, dass eine eindeutige Klassenzuordnung generierter Bilder durch das Klassifikationsmodell als Indikator für visuelle Qualität interpretiert werden kann. Diese Eigenschaft wird über die Entropie der bedingten Klassifikationsverteilung $p(y|x)$ gemessen [19, 26, 28, 93]. Zweitens wird vorausgesetzt, dass ein gutes generatives Modell in der Lage ist, eine Vielzahl unterschiedlicher Bilder zu erzeugen, die verschiedene Klassen abdecken. Dies zeigt sich in einer möglichst gleichmäßigen Verteilung der marginalen Klassifikationsverteilung $p(y)$ über alle generierten Bilder [19, 26, 28, 93]. Der Inception Score ergibt sich formal aus der exponentiellen Erwartung der KLD zwischen der bedingten Verteilung $p(y|x)$ und der marginalen Verteilung $p(y)$ und bildet somit die beiden Eigenschaft ab, die ein gutes Modell erfüllen soll. Formal gilt [26]:

$$\mathsf{IS}(G) = \exp\left(\mathbb{E}_{x\sim G}\left[D_{\mathsf{KL}}(p(y|x) \parallel p(y))\right]\right). \tag{2.37}$$

Die KLD nimmt hohe Werte an, wenn $p(y|x)$ eine niedrige Entropie aufweist, während $p(y)$ eine hohe Entropie besitzt. In der Praxis wird ein höherer Inception Score als Hinweis auf bessere Modellleistung interpretiert.

2.4.2. Fréchet-Inception-Distanz

Die FID ist ein Distanzmaß, das auch aus der synthetischen Bildgenerierung kommt. Sie bewertet nicht die Qualität einzelner Bilder, sondern die Fähigkeit des Modells, die Verteilung der Originalbilder zu reproduzieren. Dafür werden die realen und generierten Bilder zunächst mithilfe eines Feature-Extraktors in eine niedrigdimensionale und semantisch bedeutungsvollere Repräsentation gebracht. Diese ermöglicht es, die Unterschiede und Gemeinsamkeiten besser zu analysieren, da die Verteilungen der Bilder in einem Merkmalsraum verglichen werden, der auf semantisch relevante Bildstrukturen fokussiert ist. Zusätzlich ist die Berechenbarkeit der FID erst durch die niedrigere Dimensionalität der abgebildeten Daten gegeben [25]. Bei Bildern wird dafür ein vortrainiertes CNN verwendet, ursprünglich und namensgebend, das Inception Modell. Das Inception Modell ist ein Modell, welches auf dem ImageNet-Datensatz auf die Klassifikation von Bildern trainiert wurde [94]. Um die kodierte Repräsentation der Bilder zu enthalten, wird die Ausgabe des Embedding-Layers, dem letzten Layer vor dem Klassifikationskopf verwendet. Dieser liefert einen eindimensionalen Vektor, der semantisch relevante Bildmerkmale repräsentiert [95]. Die Ausgaben des Embedding-Layers können hinsichtlich aller generierten und realen Bilder als multivariate Wahrscheinlichkeitsverteilung angenommen werden. Die Ähnlichkeiten der beiden Verteilungen lassen sich über ihre statistischen Momente analysieren, da sie die charakteristischen Eigenschaften von Wahrscheinlichkeitsverteilung angeben. Der k-te Moment einer Wahrscheinlichkeitsverteilung ist definiert als $m_k = \mathbb{E}[X^k]$, und der k-te zentrale Moment als $\mu_k = \mathbb{E}[(X-\mu)^k]$ [96]. Aus Gründen der numerischen Stabilität und Berechenbarkeit werden die Verteilungen nur anhand der ersten zwei Momente verglichen. Nach dem Maximum-Entropy-Prinzip

wird angenommen, dass die multivariaten Verteilungen aus dem Embedding Layer Gaußschen Verteilungen entsprechen, da diese Verteilungen unter der Berücksichtigung der ersten zwei Momente keine zusätzlichen Annahmen enthält [97]. Die Ähnlichkeit zwischen zwei Gaußschen Verteilungen kann mithilfe der Fréchet-Distance (FD) berechnet werden [25]. Somit ist die FID definiert als die FD zwischen zwei gaußschen Approximationen und man erhält formal:

$$\mathrm{FD} = \|\mu - \mu_w\|_2^2 + \mathrm{Tr}\left(\mathbf{\Sigma} + \mathbf{\Sigma}_w - 2\left(\mathbf{\Sigma}\mathbf{\Sigma}_w\right)^{1/2}\right), \tag{2.38}$$

wobei μ und μ_w die Mittelwerte der beiden Verteilungen sind, $\mathbf{\Sigma}$ und $\mathbf{\Sigma}_w$ die entsprechenden Kovarianzmatrizen, Tr die Spur einer Matrix, also die Summe ihrer Diagonalelemente und $(\mathbf{\Sigma}\mathbf{\Sigma}_w)^{1/2}$ die Matrixwurzel des Produkts der beiden Kovarianzmatrizen darstellt [98]. Ein FID-Wert von Null bedeutet, dass die beiden Gaußschen Verteilungen identisch sind. Ein FID-Wert von Null in Bezug auf Bilder, bedeutet jedoch nicht, dass die Verteilungen der Bilder identisch sind. Das liegt daran, dass die FID lediglich auf dem ersten und zweiten Moment basiert und somit Unterschiede in höheren Momenten der Verteilung nicht abbildet. Dies wird exemplarisch im eindimensionalen Fall in Abbildung 2.9 deutlich, wo zwei unterschiedliche Verteilungen einen FID-Wert von 0 aufweisen. Die FID stellt somit lediglich eine approximative der tatsächlichen Distanz zwischen zwei Verteilungen dar. In der Praxis gilt jedoch die Annahme als gerechtfertigt, dass eine geringe FID

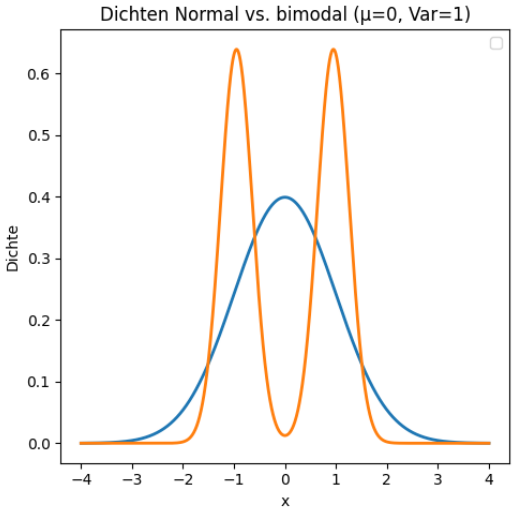

Abbildung 2.9.: Vergleich der Dichtefunktionen einer Standardnormalverteilung $\mathcal{N}(0,1)$ und einer bimodalen Mischung aus zwei Normalverteilungen mit gleicher Gesamtvarianz. Die Mischung besteht aus $0{,}5\cdot\mathcal{N}(-a,s^2)+0{,}5\cdot\mathcal{N}(+a,s^2)$ mit $a=0{,}95$ und $s^2=1-a^2=0{,}0975$, sodass die Gesamtvarianz $\mathrm{Var}(X)=s^2+a^2=1$ beträgt. Trotz unterschiedlicher Verteilungen ergibt sich ein FD von 0 da Mittelwert und Varianz identisch sind.

ein Indikator für die Qualität eines generativen Modells hinsichtlich der Ähnlichkeit der Verteilungen der generierten Bilder und der realen Bilder ist [99].

Die FID hängt, wie Chong und Forsyth [28] zeigen, von der Größe der verwendeten Stichprobe ab und weist einen systematischen Bias auf, der mit der Stichprobengröße skaliert und modellabhängig ist. Dadurch können zwei Modelle bei gleicher Qualität und begrenzter Stichprobengröße unterschiedliche FID-Werte erhalten. Eine feste Anzahl an Samples gewährleistet somit keine verlässliche Vergleichbarkeit. Um diesen Bias zu reduzieren, schlagen die Autoren vor, die FID für verschiedene Stichprobengrößen zu berechnen und anschließend eine Regression in $1/n$ durchzuführen, um den Grenzwert für $n \to \infty$ zu extrapolieren. Der so berechnete Wert wird als FID_∞ bezeichnet und stellt eine bias-reduzierte Schätzung dar [28].

2.5. Zufällige Projektionen

Zufällige Projektionen finden Anwendung im Rahmen des Johnson-Lindenstrauss-Lemmas, das ein fundamentales Resultat zur Erhaltung von Distanzen bei linearer Dimensionsreduktion darstellt. Das Lemma besagt, dass eine Menge von Punkten aus einem hochdimensionalen Raum mithilfe einer geeigneten linearen Projektionsmatrix in einen Raum niedrigerer Dimension abgebildet werden kann. Dabei bleiben die paarweisen Distanzen zwischen den Punkten mit hoher Wahrscheinlichkeit erhalten. Eine geeignet gewählte Zufallsmatrix kann dabei als Projektionsmatrix dienen. Typischerweise bestehen die Einträge einer Zufallsmatrix aus unabhängigen und identisch verteilten Zufallsvariablen. Eine normalisierte quadratische Zufallsmatrix besitzt mit hoher Wahrscheinlichkeit nahezu orthonormale Spalten und erhält dadurch die paarweisen euklidischen Abstände der Daten. Eine nicht quadratische normalisierte Zufallsmatrix ist nicht orthogonal, erfüllt jedoch mit hoher Wahrscheinlichkeit eine annähernd isometrische Abbildung. Das bedeutet, dass paarweise Distanzen bis auf einen kontrollierten Fehler erhalten bleiben [100].

Ein weiteres Anwendungsgebiet zufälliger Projektionen ist das Compressed-Sensing. Compressed-Sensing ist ein Forschungsfeld, das sich an der Schnittstelle von Informatik und Signalverarbeitung ansiedelt [101, 102]. Ziel des Compressed-Sensing ist es, dünn besetzte (sparse) Signale $x \in X \subseteq \mathbb{R}^n$ durch eine möglichst geringe Anzahl linearer Messungen $y \in Y \subseteq \mathbb{R}^m$ zu repräsentieren. Ein dünn besetzter beziehungsweise k-sparser Vektor x ist ein Vektor, der höchstens $k < n$ nichtnull Elemente besitzt. Die zugehörigen Indizes i, für die $x_i \neq 0$ gilt, sind dabei unbekannt und folgen keiner Struktur.

Da das Ziel verfolgt wird, x mit so wenig linearen Messungen wie möglich darzustellen, gilt $m < n$, sodass es sich bei der Transformation um eine Dimensionsreduktion handelt. Eine mögliche Anwendung besteht darin, $x_i \in \mathbb{R}^n$ und $i \in \mathbb{N}$ durch die Abbildung $A \in \mathbb{R}^{m \times n}$ in niedrigdimensionale Vektoren $y_i \in \mathbb{R}^m$ zu transformieren, und anschließend die sparsen Vektoren aus den niedrigdimensionalen Vektoren wieder zu rekonstruieren [103]. Dadurch kann Speicherplatz eingespart werden, da das ursprüngliche Signal mit hoher Genauigkeit rekonstruiert werden kann.

Ein alternatives Ziel besteht darin, die geometrische Struktur der Daten unter der Transformation zu bewahren. Das bedeutet, die Dimension der Matrix A wird so gewählt, dass die paarweisen Abstände zwischen Vektoren vor und nach der Abbildung erhalten bleiben. Beide Anwendungen verfolgen das Ziel einer Dimensionsreduktion mit kontrolliertem Informationsverlust. Der Informationsverlust wird dabei jedoch unterschiedlich definiert. Bei der Signalrekonstruktion steht im Vordergrund, wie genau das ursprüngliche Signal aus der niedrigdimensionalen Darstellung rekonstruiert werden kann. Bei der Strukturerhaltung hingegen beschreibt der Informationsverlust das Ausmaß, in dem sich die relativen Abstände zwischen den Vektoren durch die Transformation verändern. Es lässt sich zeigen, dass beide Zielsetzungen eng miteinander verwandt sind. Allerdings unterscheidet sich die Toleranz gegenüber Verzerrungen. Die Rekonstruktion liefert eine strikte Grenze für den zulässigen Informationsverlust. Die Strukturerhaltung hingegen erlaubt eine flexiblere Grenze, deren Ausmaß von der jeweiligen Anwendung und der akzeptierten Verzerrungstoleranz abhängt. Beide Fälle lassen sich mathematisch beschreiben durch:

$$y = Ax \quad \text{mit} \quad A \in \mathbb{R}^{m \times n}, \tag{2.39}$$

oder auch mit Berücksichtigung eines Messfehlers oder Störung in den ursprünglichen Daten:

$$y = Ax + \varepsilon, \tag{2.40}$$

mit einem Störterm $\varepsilon \in \mathbb{R}^m$ und der Projektionsmatrix A, die über alle Messungen hinweg konstant ist [104]. Die von Candès in [103] vorgestellte Restricted-Isometry-Property (RIP) garantiert eine verlustfreie Rekonstruktion dünn besetzter Signale und liefert Aussagen über die Strukturerhaltung unter der Projektion [104]. Dabei erfüllt eine Matrix $A \in \mathbb{R}^{m \times n}$ die RIP der Ordnung k, wenn es eine Konstante δ_k gibt, sodass für alle k-sparsamen Vektoren $x \in \mathbb{R}^n$ gilt [103]:

$$(1 - \delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_k)\|x\|_2^2. \tag{2.41}$$

Dabei ist $\| \ \|_2$ die euklidische Norm, δ_k die Restricted-Isometry-Konstante (RIC) für die gilt $\delta_k \in [0, 1[$. Die RIC ist die kleinste skalare Zahl, für die die Ungleichung für alle k-sparsen Vektoren x wahr ist. Sie hängt insbesondere von der Dimension m der Projektion sowie vom Sparsitätsgrad k ab.

Die RIP vergleicht die euklidischen Längen des k-sparsen Vektors x und seinem Bild y unter der Abbildung A. Dabei wird die Länge des Vektors im Bild um höchstens den relativen Faktor δ_k verändert. Auch die Veränderung des Abstands durch die Abbildung A, zwischen zwei k-sparsen Vektoren, der definiert ist als $\|x_1 - x_2\|_2^2$, lässt sich durch die RIP beschränken. Dabei setzen wir $h = x_1 - x_2$, da x_1 und x_2 k-sparse sind, folgt, dass h höchstens $2k$-sparse ist. Setzt man dies in die RIP ein, erhält man:

$$(1 - \delta_{2k})\|h\|_2^2 \leq \|Ah\|_2^2 \leq (1 + \delta_{2k})\|h\|_2^2. \tag{2.42}$$

Somit verändern sich die Abstände zwischen zwei k-sparsen Vektoren um höchstens δ_{2k}.

Soll durch die Transformation die Struktur der Daten erhalten bleiben, muss die Projektionsmatrix A eine niedrige RIC für k beziehungsweise $2k$ besitzen. Sollen die transformierten Vektoren rekonstruiert werden muss:

$$\delta_{2k} < \sqrt{2} - 1, \tag{2.43}$$

gelten. In diesem Fall ist eine eindeutige Rekonstruktion möglich, wenn die Daten nicht verrauscht sind und eine stabile Rekonstruktion, wenn sie verrauscht sind. Die Rekonstruktion erfolgt durch die Minimierung von:

$$x \;=\; \arg\min_{z \in \mathbb{R}^n} \; \frac{1}{2} \left\| Az - y \right\|_2^2 \;+\; \lambda \left\| z \right\|_1 , \tag{2.44}$$

die auch als ℓ_1-Minimierung mit Strafterm oder Lasso-Regression bezeichnet wird [105, 106]. Das Bestimmen der RIC einer gegebenen Matrix A für eine vorgegebene Sparsity k ist jedoch NP-schwer [107, 108].

Jedoch lässt sich zeigen, dass sowohl gaußsche als auch sub-gaußsche Zufallsmatrizen eine geeignete RIC besitzen [109]. Dabei handelt es sich um Matrizen, deren Einträge unabhängige und identisch verteilte sub-gaußsche Zufallsvariablen sind. Sei $A \in \mathbb{R}^{m \times n}$ eine sub-gaußsche Zufallsmatrix. Ist die Dimension der normierten Matrix $\bar{A} = \frac{1}{\sqrt{m}} A$ groß genug erfüllt sie mit hoher Wahrscheinlichkeit die RIP der Ordnung k für jedes Sparsity-Level $k \in 1, \ldots, n$ und jede gewünschte Genauigkeit $\delta \in (0, 1)$. Konkret gilt nach [110]:

$$m \geq C \delta^{-2} k \log\left(\frac{en}{k}\right) \quad \Rightarrow \quad \delta_k(\bar{A}) \leq \delta, \tag{2.45}$$

mit Wahrscheinlichkeit mindestens:

$$1 - 2\exp(-c\delta^2 m), \tag{2.46}$$

wobei C eine Konstante ist, die größer als Null ist. Diese Bedingung garantiert, dass die Länge eines k-sparsamen Vektors $x \in \mathbb{R}^n$ unter der Projektion durch $\bar{A}$ nur um höchstens δ verzerrt wird. Um zusätzlich sicherzustellen, dass auch die euklidische Distanz zwischen zwei k-sparsamen Vektoren $x_i, x_j \in \mathbb{R}^n$für$i \neq j$ nur um höchstens δ verzerrt wird, muss die Matrix $\bar{A}$ die RIP der Ordnung $2k$ erfüllen (siehe Gleichung (2.42)). Daraus folgt, dass man für ein gegebenes m den Fehler der Verzerrung, der Länge der Vektoren, als einzelne mit k oder den Abständen zwischen zwei Punkten, mit $2k$ bestimmen kann:

$$\delta \geq \sqrt{C \cdot k \cdot \log\left(\frac{en}{k}\right) \cdot \frac{1}{m}}. \tag{2.47}$$

Die Dimension von A, die hinreichend ist, um Rekonstruktion zu ermöglichen, lässt sich aus Gleichung (2.45) und Gleichung (2.43) bestimmen. Die Rekonstruktion basiert auf der RIP der Ordnung $2k$ und es muss δ_{2k} kleiner als $\sqrt{2} - 1$ gelten. Setzt man den gewünschten Wert für δ in (2.45) ein, ergibt sich die Anzahl an Messungen m, die benötigt werden, um eine Rekonstruktion mit hoher Wahr-

scheinlichkeit zu gewährleisten. Diese Schätzung für m ist eine obere Schranke für die Rekonstruktion und deshalb sehr groß. Eine untere Schranke lässt sich aus Li et al. [111] ableiten, die eine Schranke für δ_k liefern.

Theoretisch lässt sich zeigen, dass die Anzahl benötigter Messungen für eine stabile Rekonstruktion asymptotisch geringer sein kann. Foucart und Rauhut zeigen in [112], dass bereits

$$m \geq C\, k \log\left(\frac{N}{m}\right), \tag{2.48}$$

Messungen ausreichen können, um eine stabile Rekonstruktion mittels ℓ_1-Minimierung zu ermöglichen. Bei dieser Abschätzung handelt es sich um eine asymptotische Optimalitätsaussage.

Der Unterschied der Abschätzungen für m der beiden Schranken, wird im Folgenden anhand eines Beispiels illustriert. Gegeben seien Vektoren $x_i \in \mathbb{R}^n$ mit

$$n = 64 \cdot 64 \cdot 4 = 16384$$

und einer Sparsity von $k = 1000$. Eine stabile Rekonstruktion auf Basis der RIP der Ordnung $2k$ ist hinreichend gewährleistet, wenn $\delta_{2k} < \sqrt{2} - 1$ gilt. Setzt man beispielhaft $\delta_{2k} = 0{,}4$ in Gleichung (2.45) ein, ergibt sich die hinreichende Abschätzung

$$m \gtrsim C\, \delta_{2k}^{-2}\, 2k \log\left(\frac{en}{2k}\right) = C \cdot 6{,}25 \cdot 2000 \cdot \log\left(\frac{e \cdot 16384}{2000}\right) \approx 3{,}9 \cdot 10^4\, C. \tag{2.49}$$

Selbst für moderate Konstanten C liegt die so erhaltene Messzahl in einer Größenordnung, die die ursprüngliche Dimension n übersteigt und damit nicht sinnvoll ist. Dieses Beispiel verdeutlicht den stark konservativen Charakter RIP-basierter hinreichender Schranken.

Die asymptotische Schranke aus Gleichung (2.48) zeigt demgegenüber, dass die theoretisch minimale Messzahl für eine stabile Rekonstruktion wesentlich geringer sein kann. Bezogen auf das Beispiel ist die Ungleichung aus Gleichung 2.48 für $m = 4000$ und $C \leq 2.84$ erfüllt. Diese Aussage beschreibt jedoch ausschließlich die prinzipiell erreichbare Größenordnung der Messzahl und erlaubt keine konkrete numerische Abschätzung für gegebene Parameter. Der Vergleich macht deutlich, dass zwischen konstruktiven RIP-basierten Garantien und optimalen asymptotischen Resultaten eine erhebliche Lücke bestehen kann.

Zufallsprojektionen erweisen sich insgesamt als eine geeignete Methode zur Dimensionsreduktion dünnbesetzter Vektoren, auch wenn die tatsächlich benötigte Messzahl in der Praxis deutlich unterhalb strenger RIP-Grenzen liegen kann.

In vielen Anwendungen, in denen zufällige Projektionen eingesetzt werden, sind die zu verarbeitenden Signale $x \in \mathbb{R}^n$ nicht dünn besetzt [109]. Es existiert jedoch

eine verlustfreie und umkehrbare Transformation $\Phi : \mathbb{R}^n \to \mathbb{R}^n$, sodass das transformierte Signal $c = \Phi(x)$ eine dünnbesetzte Struktur aufweist. Das ursprüngliche Signal lässt sich somit als $x = \Phi^{-1}(c)$ darstellen, wobei c ein k-sparser Vektor ist. Die Transformation Φ kann beispielsweise eine Basiswechseltransformation oder auch eine allgemeine, nichtlineare, aber invertierbare Abbildung sein [104]. Beispiele für solche Transformationen sind Wavelet- oder Fourier-Transformationen, die in der Signalverarbeitung eingesetzt werden, um Signale in dünnbesetzte Signale zu transformieren.

2.6. Welfords Algorithmus zur Berechnung von Mittelwert und Kovarianz

Zur Berechnung der FID zwischen zwei Datensätzen werden der Mittelwert und die Kovarianzmatrix der beiden Datensätze benötigt. Die Standardformel für den Mittelwert einer Menge von m Vektoren $x^{(k)} \in \mathbb{R}^n$ mit $k \in \{1, \ldots, m\}$, wobei $x^{(k)}$ den k-ten Vektor des Datensatzes bezeichnet, ist in vektorisierter Form definiert als [113, 114]:

$$\bar{x} = \frac{1}{m} \sum_{k=1}^{m} x^{(k)}, \tag{2.50}$$

und die Kovarianzmatrix $\Sigma \in \mathbb{R}^{n \times n}$ definiert als [113]:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathrm{cov}(x_1, x_1) & \ldots & \mathrm{cov}(x_1, x_n) \\ \vdots & \ddots & \vdots \\ \mathrm{cov}(x_n, x_1) & \ldots & \mathrm{cov}(x_n, x_n) \end{bmatrix}, \tag{2.51}$$

wobei $x_i^{(k)}$ die i-te Komponente des k-ten Vektors ist und die Kovarianz zwischen den Komponenten i und j über alle Vektoren berechnet wird. Die aus einem Datensatz stammende Kovarianz zwischen x_i und x_j ist definiert als:

$$\mathrm{cov}(x_i, x_j) = \frac{1}{m-1} \sum_{k=1}^{m} (x_i^{(k)} - \bar{x}_i)(x_j^{(k)} - \bar{x}_j). \tag{2.52}$$

Obwohl diese Formel in zahlreichen Anwendungsbereichen nützlich ist, weist sie Einschränkungen auf. Ein wesentlicher Nachteil besteht darin, dass der gesamte Datensatz zum selben Zeitpunkt im Speicher vorliegen muss, da für die Berechnung des Mittelwerts und der Kovarianz zwei vollständige Durchläufe über die Daten erforderlich sind [114].

Welford löst dieses Problem in [115], indem er einen Online-Algorithmus zur Berechnung der Varianz und des Mittelwertes einführt. Ein Online-Algorithmus ist ein Algorithmus, der eingehende Daten schrittweise verarbeitet, ohne dass der vollständige Datensatz vorliegen muss. Stattdessen werden eingehende

Datenströme sequenziell verarbeitet [116, 117]. Der Algorithmus ist so konzipiert, dass der Datensatz nur einmal durchlaufen wird und nur die einzelnen Mittelwerte und Kovarianzen gespeichert und sequenziell aktualisiert werden. Es ergibt sich die Formel für das Update des Mittelwertes:

$$\bar{x}^{(k+1)} = \bar{x}^{(k)} + \frac{1}{k+1}(x^{(k+1)} - \bar{x}^{(k)}), \tag{2.53}$$

wobei $\bar{x}^{(k)}$ der Mittelwert zum Zeitpunkt des k-ten Vektors ist [115]. Für die Kovarianz wird im Online-Verfahren nicht direkt die Kovarianzmatrix $\mathbf{\Sigma}$ aktualisiert, sondern eine akkumulierte Streumatrix $\mathbf{S}$. Aus $\mathbf{S}$ ergibt sich die Kovarianzmatrix erst nach Abschluss aller Updates durch eine Skalierung. Sei dazu $\mathbf{S}^{(k)}$ die Streumatrix nach k Beobachtungen. Für das Update gilt [114, 115]:

$$\mathbf{S}^{(k+1)} = \mathbf{S}^{(k)} + \left(x^{(k+1)} - \bar{x}^{(k)}\right)\left(x^{(k+1)} - \bar{x}^{(k+1)}\right)^{\top}. \tag{2.54}$$

Nach m Vektoren erhält man daraus die Stichprobenkovarianz als:

$$\mathbf{\Sigma} = \frac{1}{m-1}\mathbf{S}^{(m)}. \tag{2.55}$$

Die vorgestellten Update-Formeln ermöglichen eine speichereffiziente und numerisch stabile Berechnung der Mittelwerte und Kovarianzmatrix, ohne dass der gesamte Datensatz gleichzeitig vorliegen muss [115]. Dies ist insbesondere bei großen Datenmengen von Vorteil. Vor allem für das Berechnen der FID stellt Welfords Algorithmus eine praktikable und robuste Alternative dar, da die extrahierten Features nur zum Zeitpunkt des Updates benötigt werden und danach verworfen werden können.

3. Methode

In diesem Kapitel wird die Fréchet-Radar-Distance (FRD) vorgestellt und begründet, warum sie für die Bewertung der Qualität synthetischer Radar-Daten geeignet ist. Zusätzlich werden die parametrisch gesteuerten Pseudoradar-Generatoren vorgestellt, die in den Experimenten verwendet werden, um das Distanzmaß zu validieren. Hierbei stützt sich dieses Kapitel auf die im vorherigen Abschnitt beschriebenen theoretischen Grundlagen, die die Basis für die Entwicklung und Bewertung der vorgestellten Methoden bilden.

3.1. FRD für Radar-Daten

Die FID ist ein Distanzmaß zur Qualitätsbewertung synthetisch erzeugter Bilder (vgl. Abschnitt 2.4.2). Das Distanzmaß misst jedoch nur den Abstand zwischen zwei Bilddatensätzen (siehe Abschnitt 2.4.2). Die Qualitätsbewertung von synthetischen Daten erfolgt indirekt, indem statistische Unterschiede zwischen realen und synthetisch erzeugten Daten gemessen werden. Die Berechnung der Distanz basiert auf einem vortrainierten Feature-Extraktor. Dieser transformiert die Bilder in einen semantisch aussagekräftigen Merkmalsraum [118]. In diesem Raum werden die Mittelwerte und Kovarianzmatrizen der realen und synthetischen Daten geschätzt. Der Abstand zwischen beiden Verteilungen wird anschließend mithilfe der Fréchet-Distanz berechnet (siehe Gleichung 2.38).

Die Transformation der Daten durch den Feature-Extraktor erfüllt bei der Berechnung der FID zwei zentrale Funktionen. Zum einen überführt sie die Bilddaten in einen Merkmalsraum, in dem semantisch relevante Strukturen expliziter repräsentiert sind. Zum anderen reduziert sie die Dimensionalität der Daten. Die Dimensionsreduktion ist notwendig, da die Dimension ausschlaggebend für die Zeitkomplexität der Kovarianzmatrixschätzung ist. Befinden sich die Daten in einem Raum mit Dimension d, so besitzt die Kovarianzmatrix die Form $C \in \mathbb{R}^{d \times d}$. Die numerische Stabilität der Schätzung der Kovarianzmatrix hängt von ihrer Dimensionalität und der Anzahl der gegebenen Datenpunkte n ab [119]. Gilt für die Schätzung $n < d$, besitzt die Kovarianzmatrix höchstens den Rang $n - 1$ [119–121]. Die Kovarianzmatrix hat dadurch keinen vollen Rang und ist somit schlecht konditioniert. Weil die FID unter anderem Matrixmultiplikationen und die Berechnung der Matrixwurzel der Kovarianzmatrizen erfordert, führt eine schlechte Konditionierung dieser Matrizen zu Instabilitäten in der FID-Berechnung. Selbst für $n = d$ ist die Schätzung stark verrauscht und numerisch instabil, was

die Aussagekraft der FID beeinträchtigt [28, 120]. Damit wird die Dimensionalität des Merkmalsraums zu einem zentralen Engpass der Methode, da sie die minimal erforderliche Stichprobengröße für eine stabile Schätzung bestimmt. Zusätzlich beeinflusst die Dimensionalität d den Rechenaufwand der FD-Berechnung (siehe Gleichung 2.38). Insbesondere trägt die Berechnung der Matrixwurzel einer $d \times d$-Matrix mit einer Zeitkomplexität von $\mathcal{O}(d^3)$ wesentlich zu den Gesamtkosten bei. Sowohl der Stichprobenbedarf als auch der Rechenaufwand skalieren somit direkt mit der Dimensionalität des Feature-Vektors. Die Verwendung eines Feature-Extraktors mit kontrollierter Ausgabedimension ist daher nicht nur aus inhaltlichen, sondern auch aus numerischen und rechentechnischen Gründen entscheidend für eine stabile und effiziente FID-Berechnung. Entsprechend sind diese Aspekte auch bei der Definition einer FRD zu berücksichtigen.

Zunächst werden die Faktoren betrachtet, die das Inception-Netz als Feature-Extraktor für die Berechnung der FID etabliert haben. Ein zentraler Faktor ist die Festlegung der zugrundeliegenden Lernaufgabe des Feature-Extraktors. Das Inception-Netz wird vor seiner Verwendung als Feature-Extraktor auf die Klassifikation von Bildern trainiert. Dieser Ansatz beruht auf der Annahme, dass Merkmalsrepräsentationen, die für die Klassifikation geeignet sind zugleich eine hohe Trennschärfe zur Unterscheidung visueller Inhalte besitzen. Ein weiterer Faktor betrifft die Einigung auf den ImageNet-Datensatz als Trainingsgrundlage. Durch seine große inhaltliche Vielfalt, in Kombination mit einem überwachten Klassifikationstraining, entsteht ein Feature-Extraktor, der ein breites Spektrum visueller Strukturen abbildet und sich für den Vergleich von Bildverteilungen eignet. Ein dritter Faktor ist die Standardisierung des Eingabeformats. Das Inception-Netz verwendet ein fest definiertes Eingabeformat, auf das Bilder unterschiedlicher Auflösung vor der Verarbeitung abgebildet werden. Diese Vereinheitlichung ermöglicht eine konsistente Merkmalsextraktion, unabhängig von der ursprünglichen Bildauflösung. In ihrer Kombination haben diese Faktoren dazu geführt, dass das Inception-Netz als Feature-Extraktor für die FID verwendet wird. Somit entspricht die Wahl den etablierten Konventionen. Sie ist jedoch nicht theoretisch begründet, da alternative Formulierungen ebenfalls möglich sind.

Für Radar-Daten existieren bislang keine vergleichbaren Konventionen zur Wahl eines vortrainierten Feature-Extraktors. Dies ist unter anderem auf die starke Domänenspezifik von Radar-Daten zurückzuführen. Bei Radar-Daten führen schon geringe Unterschiede in der Sensorarchitektur, im Frequenzband, oder im Anwendungsszenario zu signifikanten Verteilungsverschiebungen [50]. Zusätzlich fehlen öffentlich zugängliche, standardisierte Radar-Datensätze in ausreichender Größe, die das Training eines universellen Feature-Extraktors ermöglichen würden [37, 50, 122]. Des Weiteren lässt sich für Radar-Daten kein eindeutig definiertes Trainingsziel für einen Feature-Extraktor festlegen. Im Gegensatz zu Bildern, in denen Objekte als vollständige und konsistente Strukturen erscheinen, liefern Radar-Punktwolken lediglich fragmentierte Reflexionen einzelner Objektteile. Da diese Reflexionen stark szenen- und sensorabhängig sind, lässt sich keine stabile Zuordnung zu festen Objektklassen definieren. Dadurch ist ein klassifikationsbasiertes Trainingsziel für Radar-Daten ungeeignet. Alternative Trainingsziele müssten an

spezifische Aufgaben oder Szenarien gekoppelt werden, was die Entwicklung eines domänenübergreifend einsetzbaren, vortrainierten Feature-Extraktors erheblich einschränkt. Diese Punkte motivieren den Einsatz eines trainingsunabhängigen Feature-Extraktors, der ohne domänenspezifisches Trainingsziel, annotierte Datensätze und sensorabhängiges Vortraining auskommt.

Aus diesem Grund wird im Folgenden ein trainingsunabhängiger Ansatz zur Berechnung der FRD vorgestellt der auf zufälligen Projektionen, als Feature-Extraktor (siehe Abschnitt 2.5) basiert. Da zufällige Projektionen eine dünnbesetzte Darstellung der Radar-Daten erfordern, werden diese zunächst in ein geeignetes Format übertragen. Dafür werden die Radar-Punktwolken durch ein verlustfreies Diskretisierungsverfahren in eine rasterbasierte Struktur überführt, die eine einheitliche und strukturierte Eingabeform bereitstellt. Anschließend wird diese diskretisierte Darstellung, mittels zufälliger Projektionen in einen niedrigdimensionalen Merkmalsraum abgebildet. In diesem Merkmalsraum lassen sich statistische Kenngrößen effizient bestimmen. Auf Basis der statistischen Kenngrößen wird die FD als Distanzmaß zwischen zwei Datensätzen berechnet. Die Abfolge dieser Schritte bildet eine geschlossene Pipeline zur Bewertung synthetischer Radar-Daten, die in den folgenden Abschnitten detailliert beschrieben wird.

3.1.1. Diskretisierung von Radar-Punktwolken

Zur Überführung von Radar-Punktwolken, in eine für die weitere Verarbeitung geeignete Form, wird ein Diskretisierungsverfahren eingesetzt. Ziel ist es, die unregelmäßig verteilten Punktdaten ohne Informationsverlust, in eine strukturierte, rasterbasierte und sparse Darstellung zu überführen. Dafür wird für den Wertebereich der Punktkoordinaten ein regelmäßiges Gitter definiert. Die Gittergröße wird vorab festgelegt und orientiert sich an der maximalen Anzahl von Punkten, die das Radarsystem erfassen kann. Diese ist durch physikalische und technische Eigenschaften des Sensors begrenzt. Auf Basis dieser Obergrenze, kann eine geeignete Gitterauflösung gewählt werden, die eine sparse Belegung des Rasters sicherstellt. Anschließend wird jeder Punkt genau einer Gitterzelle und jeder Gitterzelle höchstens einen Punkt zugeordnet. Die Zuordnung erfolgt anhand der minimalen euklidischen Distanz, zwischen dem Punkt und dem Mittelpunkt der jeweiligen Gitterzelle.

Dieses Zuordnungsproblem lässt sich als Linear-Sum-Assignment-Problem (LSAP) formulieren bei dem eine Menge von Objekten einer Menge von Positionen unter Berücksichtigung individueller Kosten zugewiesen werden [123]. Im klassischen LSAP werden N Aufgaben N Agenten zugewiesen, wobei jede Zuweisung mit Kosten verbunden ist. Für die Zuweisung von Punkten zu Gitterzellen werden maximal $k \ll N$ Punkte einer Menge von N Gitterzellen zugewiesen. Die Kosten der Zuweisung eines Punktes i zur Gitterzelle j, werden durch die euklidische Distanz zwischen dem Punkt und dem Mittelpunkt der Zelle beschrieben und als

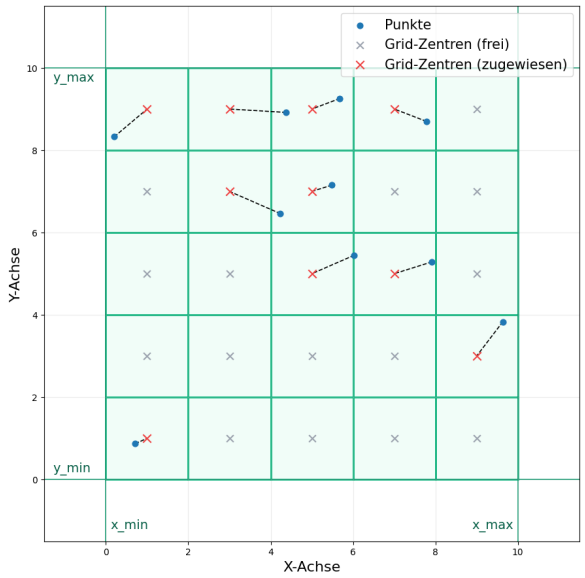

Abbildung 3.1.: Visualisierung der optimalen Zuordnung von Punkten zu Grid-Zentren mit 10 Punkten und einer Grid-Größe von 5 x 5 und einem vorgegebenen Wertebereich

c_{ij} definiert. Das lineare Optimierungsproblem lässt sich somit definieren als:

$$\min_{x_{ij}} \sum_{i=1}^{k} \sum_{j=1}^{N} c_{ij} \cdot x_{ij}, \tag{3.1}$$

unter den Nebenbedingungen:

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall i \in \{1, \ldots, k\} \tag{3.2}$$

$$\sum_{i=1}^{k} x_{ij} \leq 1 \quad \forall j \in \{1, \ldots, N\} \tag{3.3}$$

$$x_{ij} \in \{0, 1\}. \tag{3.4}$$

Dabei gibt $x_{ij} = 1$ an, dass Punkt i der Gitterzelle j zugewiesen wurde. Ziel ist es, die Gesamtkosten der Zuordnung zu minimieren (siehe Abbildung 3.1). Die exakte Lösung dieses Problems kann mit dem Ungarischen Algorithmus in $\mathcal{O}(N^3)$ berechnet werden. Zur Reduktion der Rechenkomplexität kann ein heuristischer Ansatz verwendet werden. Dabei werden zunächst alle Gitterzellen identifiziert, denen genau ein Punkt zugeordnet werden kann, und diese Zuordnungen direkt festgelegt. Danach wird ein lineares Zuweisungsproblem mit den noch übrig gebliebenen Grid-Zellen und Punkten gelöst. Dieser Ansatz liefert in der Praxis eine ausreichende Qualität, garantiert jedoch keine optimale Lösung. Die Speicherung der Punkte im Gitter erfolgt durch eine binäre Belegungsinformation in der ersten

Dimension. Ergänzend werden die relativen Abweichungen vom Zellmittelpunkt, in x und y in der zweiten und dritten Dimension jeder Gitterzelle gespeichert. Die Speicherung der Abweichungen ermöglicht eine verlustfreie Rekonstruktion der Punktwolke aus der Gitterdarstellung (vgl. Abb. 3.2). Liegen für einen detektierten Punkt weitere radarspezifischen Informationen vor, werden diese ebenfalls in der jeweiligen Gitterzelle gespeichert. Dieses Verfahren erhält die räumliche Struktur der Punktwolke und erzeugt zugleich eine diskrete, rasterbasierte Darstellung. Die resultierende sparse Belegung des Gitters, bildet die Grundlage für die Anwendung zufälliger Projektionen als Feature-Extraktor.

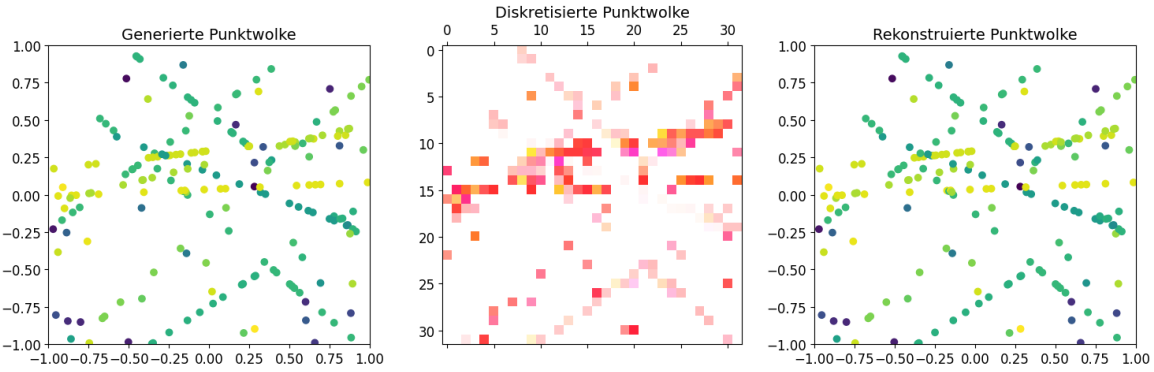

Abbildung 3.2.: Visualisierung einer mit Algorithmus 3 erzeugten Punktwolke. Zusätzlich dargestellt sind das diskretisierte Datenformat sowie die verlustfreie Rekonstruktion der ursprünglichen Punktwolke aus der Diskretisierung. Die Farbe im diskretisierten Datenformat entsteht durch das Darstellen der Gitter als RGB-Bild.

3.1.2. Zufällige Projektion als Feature-Extraktor

Aufbauend auf der zuvor beschriebenen Diskretisierung der Radar-Punktwolken, wird im Folgenden die Verwendung, zufälliger Projektionen als Feature-Extraktor betrachtet [37]. Zufällige Projektionen stellen eine trainingsunabhängige Alternative zu vortrainierten, domänenspezifischen Feature-Extraktoren dar. Sie ermöglichen die Abbildung hochdimensionaler dünnbesetzter Daten, in einen niedrigdimensionalen Merkmalsraum und sind deshalb für Radar-Daten geeignet [112]. Für die Abbildung wird die rasterbasierte Gitterdarstellung der Radar-Punktwolken zunächst zu einem d-dimensionalen Vektor abgeflacht. Die Abbildung erfolgt danach formal, durch die Multiplikation der d-dimensionalen Eingabedaten, mit einer zufälligen Projektionsmatrix $\mathbf{R} \in \mathbb{R}^{d \times \ell}$, wobei $\ell \ll d$ gilt. Die Einträge der Projektionsmatrix sind unabhängig und identisch standardnormalverteilt. Dadurch ist die Projektion unabhängig von der zugrundeliegenden Datenverteilung [124, 125]. Die zentrale Voraussetzung zur effizienten Anwendung der zufälligen Projektion ist die dünnbesetzte Darstellung der Eingabevektoren. Diese wird bei der FRD-Berechnung durch das Diskretisierungsverfahren sichergestellt. Die Sparsity

```python
def frechet_radar_distance(data_reference, data_comparison,
    data_dim, feature_dim):
    rp = RandomProjection(data_dim, feature_dim)
    ref_stat = OnlineStats(feature_dim)
    comp_stat = OnlineStats(feature_dim)
    for ref_data, comp_data in zip(data_reference, data_comparison):
        ref_latent = rp.forward(ref_data)
        comp_latent = rp.forward(comp_data)

        ref_stat.update(ref_latent)
        comp_stat.update(comp_latent)
    return fd(ref_stat, comp_stat)
```

Python Code 3.1: Berechnung der FRD [37]

erlaubt es effektiv, die Dimensionalität der Daten zu reduzieren. Unter geeigneten Bedingungen hinsichtlich der Dünnbesetztheit der Eingangsdaten und der Dimension des Projektionsraums, lassen sich die ursprünglichen Daten aus den transformierten Daten rekonstruieren (siehe Abschnitt 2.5). Zusätzlich führen die Projektionen lediglich zu einer kontrollierten Verzerrung der Unterschiede zwischen zwei Vektoren vor und nach der Abbildung. Das bedeutet, dass Gitter, die sich vor der Projektion unterscheiden, auch im projizierten Merkmalsraum unterscheidbar bleiben. Sind Gitter vor der Abbildung ähnlich, liegen sie auch im Merkmalsraum nahe beieinander (siehe Abschnitt 2.5). In Kombination mit dem verlustfreien Diskretisierungsverfahren stellen zufällige Projektionen somit einen geeigneten, trainingsunabhängigen Feature-Extraktor für Radar-Daten dar.

3.1.3. Zusammenführung der Schritte zur FRD-Berechnung

Die Berechnung der FRD basiert somit auf der klar definierten Verarbeitungskette. Werden zwei Radar-Datensätze miteinander verglichen, geht man wie folgt vor: Zunächst werden die Radar-Punktwolken mithilfe des Diskretisierungsverfahrens in eine strukturierte, rasterbasierte und sparse Darstellung überführt. Diese Darstellung wird anschließend zu einem Vektor abgeflacht. Dieser dient als Eingabe für den Feature-Extraktor. Der Feature-Extraktor wird mithilfe von zufälligen Projektionen umgesetzt und nutzt damit die sparse Darstellung nach der Diskretisierung aus. Anschließend werden in dem durch die zufälligen Projektionen geschaffenen Merkmalsraum die Mittelwerte und die Kovarianzmatrizen der beiden Datensätze geschätzt. Die Schätzung kann dabei inkrementell mithilfe des Welford-Algorithmus (vgl. Abschnitt 2.6) umgesetzt werden. Das ermöglicht eine speichereffiziente Berechnung, da die einzelnen Merkmalsvektoren nicht vollständig gespeichert werden müssen. Auf Basis der geschätzten Mittelwerte und Kovarianzmatrizen wird abschließend die FD zwischen den beiden Merk-

malsverteilungen berechnet (siehe Gleichung 2.38). Die beschriebene Pipeline ermöglicht somit eine konsistente und trainingsunabhängige Bewertung synthetischer Radar-Daten. Listing 3.1 ergänzt die textuelle Beschreibung um eine Referenzimplementierung. Dadurch werden insbesondere Iterationsschema, Parametrisierung und die inkrementelle Schätzung der Statistikgrößen eindeutig nachvollziehbar.

3.1.3.1. Bias-Korrektur und Grenzwertschätzung der FRD

Die FRD wird auf Basis geschätzter Mittelwerte und Kovarianzmatrizen berechnet. Da diese statistischen Größen bei endlicher Stichprobengröße nur näherungsweise bestimmt werden können, ist die resultierende FRD in Abhängigkeit von der Anzahl der verwendeten Samples n systematisch verzerrt. Speziell wirken sich Schätzfehler der Kovarianzmatrizen aufgrund der nichtlinearen Struktur der FD direkt auf den Distanzwert aus [28].

Um diesen Stichproben-Bias zu reduzieren, kann eine reduziert verzerrte Schätzung FRD_∞ bestimmt werden. Dazu wird die FRD für mehrere Stichprobengrößen n berechnet und anschließend als Funktion von $1/n$ modelliert. Durch Extrapolation des Grenzwerts für $n \to \infty$ lässt sich der asymptotische, bias-freie Distanzwert abschätzen. Konkret wird hierfür eine lineare Regression der Form

$$\text{FRD}(n) = a + \frac{b}{n}, \tag{3.5}$$

durchgeführt, wobei der Achsenabschnitt a die gesuchte Grenzwertschätzung FRD_∞ darstellt. Dieses Vorgehen folgt dem Ansatz von Chong und Forsyth [28] und ermöglicht eine faire Vergleichbarkeit zwischen Modellen, da der Einfluss der Stichprobengröße explizit berücksichtigt wird (siehe Abschnitt 2.4.2).

Ein wesentlicher Vorteil dieses Ansatzes besteht darin, dass die Berechnung der FRD_∞ nur geringe Zusatzkosten verursacht. Die während des sequenziellen Durchlaufens mittels Welford-Algorithmus inkrementell berechneten Mittelwerte und Kovarianzmatrizen, können direkt für die Auswertung verschiedener $\text{FRD}(n)$ Werte wiederverwendet werden.

3.2. Parametrisierte Pseudoradar-Generatoren

Damit die FRD zur Bewertung synthetisch erzeugter Radar-Daten eingesetzt werden kann, muss das Distanzmaß zunächst validiert werden. Dafür muss nachgewiesen werden, dass Unterschiede und Gemeinsamkeiten zwischen verschiedenen Verteilungen zuverlässig identifiziert werden können [126]. Die Validierung dieser Eigenschaft ist jedoch mit mehreren methodischen Herausforderungen verbunden. Eine zentrale Schwierigkeit besteht darin, dass für echte Radar-Daten keine eindeutig definierte Referenzmetrik existiert, die als objektiver Vergleichsmaßstab

für die Bewertung der Qualität, der synthetischen Daten herangezogen werden kann [21]. Dadurch ist es nicht möglich, die FRD zwischen zwei verschiedenen Radar-Datensätzen mit einem gesichert richtigen Referenzwert zu vergleichen.

Algorithmus 3 Pseudoradar-Generator [37]

Input: $\lambda_{\text{Linien}},\ \lambda_{\text{Punkte/Linie}},\ \lambda_{\text{Clutter}}$
Output: Punktwolke
points = [ ]
sample $\mathcal{N}_L \sim \text{Poisson}(\lambda_{\text{Linien}})$
for $i = 1, \ldots, \mathcal{N}_L$ **do**
    sample $\mathcal{N}_p \sim \text{Poisson}(\lambda_{\text{Punkte/Linie}})$
    sample $m, t \sim \mathcal{U}([-1, 1])$
    for $k = 1, \ldots, \mathcal{N}_p$ **do**
        sample $x \sim \mathcal{U}([0, 1])$
        points.append$(x,\ m \cdot x + t)$
sample $\mathcal{N}_C \sim \text{Poisson}(\lambda_{\text{Clutter}})$
points.extend$(x_1, \ldots, x_{\mathcal{N}_C} \sim \mathcal{U}([-1, 1]))$
return points

Diese Herausforderung motiviert den Einsatz parametrischer Pseudoradar-Generatoren [21]. Diese Generatoren ermöglichen die Schaffung einer kontrollierten Testumgebung, in der beliebig viele Datenpunkte aus einer, durch Parameter definierten Verteilung erzeugt werden können [127]. Dies erlaubt es, das Verhalten von Distanzmaßen, wie der FRD, systematisch zu untersuchen. Insbesondere kann experimentell gezeigt werden, dass das Bewertungsmaß bei identischen Parametern und damit identischen zugrundeliegenden Verteilungen mit wachsender Stichprobengröße gegen Null konvergiert, während die Distanz bei unterschiedlichen Parametern und somit unterschiedlichen Verteilungen gegen einen von Null verschiedenen Grenzwert konvergiert. Dieses Verhalten ist eine notwendige Voraussetzung, für die Aussagekraft und Zuverlässigkeit der FRD. Beim Untersuchen der Konvergenz spielt nicht nur der Wert an sich eine Rolle, sondern auch die Anzahl der benötigten Datenpunkte und somit die Geschwindigkeit der Konvergenz. Um einen aussagekräftigen Bezug zu echten Radar-Daten zu gewährleisten, müssen die Generatoren so konzipiert sein, dass die erzeugte Verteilung, die strukturellen und stochastischen Charakteristika realer Radar-Daten möglichst präzise nachbildet.

Radar-Daten weisen, wie in Abschnitt 2.1 beschrieben, spezifische Eigenschaften auf, die maßgeblich vom gewählten Datenformat abhängen. Die in dieser Arbeit verwendeten parametrischen Pseudoradar-Generatoren erzeugen Radar-Punktwolken, deren Koordinaten durch die Laterale und Longitudinale Distanz beschrieben sind. Reale Radar-Punktwolken besitzen charakteristische Merkmale sowohl struktureller als auch zufälliger Natur. Einerseits ordnen sich detektierte Punkte entlang geometrisch strukturierter Muster an, die die physische Geometrie reflektierender Objekte widerspiegeln. Dazu zählen lineare oder gekrümmte Strukturen, etwa durch Leitplanken oder Fahrbahnmarkierungen, sowie kompakte

geometrische Formen, wie rechteckartige Punktanordnungen, die durch Fahrzeuge oder Gebäude entstehen. Andererseits enthalten Radar-Punktwolken Streuungen, die sogenannten Clutter-Punkte. Diese entstehen durch Umgebungsrauschen oder Reflexionen, an nicht relevanten Objekten und sind meist zufällig verteilt. Obwohl sie die Interpretation der Daten erschweren, stellen sie ein wesentliches Merkmal realer radarbasierter Szenarien dar und müssen bei der Modellierung berücksichtigt werden [45].

Aus diesen Eigenschaften lässt sich ein grundlegender Algorithmus für einen parametrisierten Pseudoradar-Generator ableiten, wie in Algorithmus 3 dargestellt. Der Algorithmus beschreibt dabei ein parametrisches Verfahren zur Generierung synthetischer Punktwolken in $\mathbb{R}^2$, welches zentrale Merkmale realer Radar-Daten widerspiegelt. Die erzeugten Punktwolken bestehen aus zwei Komponenten. Einerseits aus strukturierten Reflexionsmustern, die durch geometrisch geordnete Punktverteilungen entlang linearer Funktionen modelliert werden, andererseits aus stochastisch verteilten Clutter-Punkten, die Umgebungsrauschen und Mehrwegeeffekte simulieren [45]. Diese Kombination ermöglicht die Erzeugung synthetischer Punktwolken, die die strukturellen und stochastischen Eigenschaften realer Radar-Daten abstrahiert wiedergeben. Somit lassen sich typische Merkmalsverteilungen nachbilden, ohne jedoch deren physikalische Reflexionsmuster zu reproduzieren. Das Verfahren arbeitet dabei mithilfe von Poisson-Verteilungen, zur stochastischen Modellierung der Anzahl an Linien (λ_{Linien}), der Anzahl an Punkten pro Linie ($\lambda_{\text{Punkte/Linie}}$), sowie der Clutter-Intensität (λ_{Clutter}). Diese Parametrisierung ermöglicht eine flexible Kontrolle über die Komplexität und Dichte der generierten Punktwolken. Dazu kommt, dass diese Parametrisierung es erlaubt, die erwartete Anzahl an Punkten zu bestimmen, die gegeben ist durch:

$$\mathbb{E}[\text{Anzahl Punkte}] = \lambda_{\text{Punkte/Linie}} \cdot \lambda_{\text{Linien}} + \lambda_{\text{Clutter}}. \tag{3.6}$$

Zusätzlich können die generierten Punktwolken als mehrdimensionale Zufallsvariable interpretiert werden. Dabei gilt, dass alle Realisierungen dieser Zufallsvariable unabhängig und identisch verteilt (i.i.d.) sind. Diese Eigenschaft gilt nicht nur für einzelne Punktwolken, sondern auch für eine Menge von Realisierungen, einer beliebigen Größe. Daraus entsteht die Möglichkeit, zwei Datensätze zu erzeugen die unabhängig voneinander sind. Die i.i.d.-Annahme ist von zentraler Bedeutung für statistische Analysen und ML [128], da sie die Grundlage für viele theoretische Garantien bildet. Dazu zählt unter anderem die Validierung durch unabhängige Testdaten. Darüber hinaus ist der Generator modular aufgebaut und kann durch zusätzliche geometrische Strukturen wie Rechtecke, gekrümmte Kurven oder segmentierte Flächen erweitert werden, um spezifische Szenarien und Objekttypen gezielt zu modellieren.

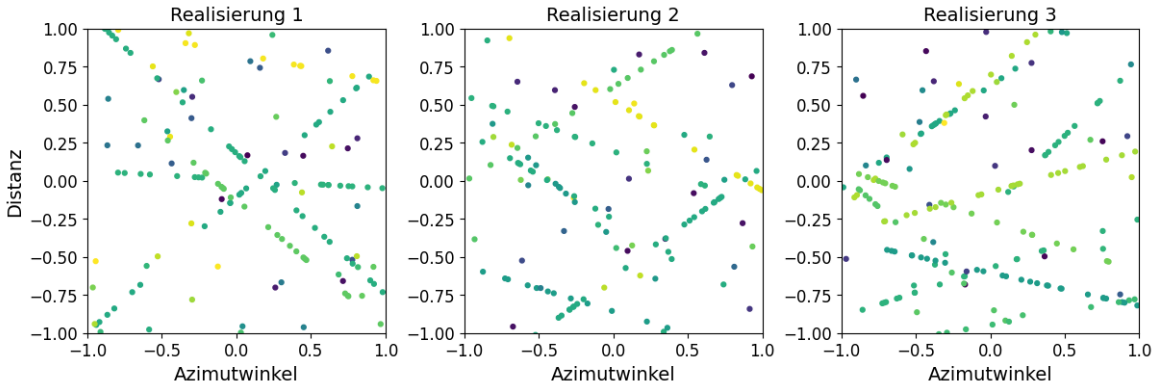

Abbildung 3.3.: Drei verschiedene Realisierungen der generierten Punktwolken mithilfe des Algorithmus 3 mit den Parametern $\lambda_{\text{Linien}} = 5$, $\lambda_{\text{Punkte/Linie}} = 20$ und $\lambda_{\text{Clutter}} = 50$. Die Farbe der Punkte entsteht für jede Linie zufällig beim Generieren der Punktwolke.

3.2.1. Log-Likelihood-Berechnung für den Pseudoradar-Generator

Ein weiterer Vorteil des Designs besteht darin, dass die Anzahl der Linien, die Punkte pro Linie, sowie die Clutter-Punkte explizit, durch die Parameter λ_{Linien}, $\lambda_{\text{Punkte/Linie}}$ und λ_{Clutter} gesteuert werden. Da diese Größen aus unabhängigen Poisson-Verteilungen stammen, kann die Likelihood einer gegebenen Punktwolke bestimmt werden, sofern die Anzahl der Linien, die Punkte auf den Linien und die Clutter-Punkte bestimmt werden können. Da die vom Generator erzeugten Punkte auf einer Linie, exakt auf dieser liegen, lässt sich ein einfacher, heuristischer Algorithmus entwickeln, um Linien und die zugehörigen Punkte zu identifizieren. Dabei ist anzumerken, dass diese Annahme eine bewusste Vereinfachung von Radar-Daten darstellt. Bei den verwendeten Pseudoradar-Generatoren liegen die erzeugten Punkte exakt auf idealisierten Linien, sodass keine zusätzliche Streuung um die Linien modelliert wird. Diese Eigenschaft ist nicht als realistische Approximation realer Radar-Daten zu verstehen, sondern dient der kontrollierten Analyse der Statistiken des Generators.

Das Verfahren basiert darauf, iterativ Punktpaare auszuwählen und durch diese, eine potenzielle Linie zu definieren. Anschließend wird überprüft, wie viele weitere Punkte einen niedrigeren Abstand als den Schwellenwert τ von der Linie aufweisen. Werden mehr als $N_{\min}$ Punkte einer Linie zugeordnet, wird die Linie akzeptiert (siehe Algorithmus 4). Der Algorithmus ist heuristisch, da Punkte, die einer identifizierten Linie zugeordnet wurden, aus der Punktmenge entfernt werden. Deshalb werden nicht alle möglichen Linienkombinationen untersucht. Dadurch wird die Rechenkomplexität reduziert, jedoch keine optimale Lösung garantiert. Mithilfe des Algorithmus kann die geschätzte Anzahl der Linien n_L, die Anzahl der Punkte auf Linie j $n_{p,j}$ und die Anzahl der Clutter-Punkte n_C bestimmt werden. Durch diese Werte lässt sich die Log-Likelihood einer Punktwolke unter dem

aktuellen Generator berechnen. Da alle drei Parameter durch den Generator unabhängig und Poisson-verteilt gezogen werden, ergibt sich die Log-Likelihood aus der Summe der einzelnen Poisson-Log-Likelihood-Terme. Für eine Zufallsvariable $X \sim \mathrm{Poisson}(\lambda)$ ist die Poisson-Verteilung gegeben durch:

$$P(X = k \mid \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}, \tag{3.7}$$

woraus sich die Log-Likelihood für eine Beobachtung k ergibt:

$$\log P(X = k \mid \lambda) = k \log \lambda - \lambda - \log(k!). \tag{3.8}$$

Damit die Anzahl der Punkte pro Linie in der Berechnung der Log-Likelihood nicht dominiert, wird die Summe über die Anzahl der Punkte pro Linie skaliert. Dadurch ergibt sich die vollständige Log-Likelihood einer Punktwolke unter den geschätzten Zählvariablen und den Parametern des Generators $\lambda_{\text{Linien}}, \lambda_{\text{Punkte/Linie}}, \lambda_{\text{Clutter}}$:

$$\begin{aligned} \mathcal{L}_{\text{log}} = &\log P(X = n_L \mid \lambda_{\text{Linien}}) \\ &+ \frac{1}{n_L} \sum_{j=1}^{n_L} \log P(X = n_{p,j} \mid \lambda_{\text{Punkte/Linie}}) \\ &+ \log P(X = n_C \mid \lambda_{\text{Clutter}}), \end{aligned} \tag{3.9}$$

Die Berechnung der Log-Likelihood ermöglicht einen quantitativen Vergleich zwischen Generatoren mit unterschiedlichen Parametern. Eine höhere Log-Likelihood weist darauf hin, dass die Daten unter dem jeweiligen Generator wahrscheinlicher sind. Es ist jedoch zu beachten, dass die Berechnung der Log-Likelihood auf der Annahme beruht, dass die beobachteten Punktwolken tatsächlich durch den zugrundeliegenden Pseudoradar-Generator erzeugt wurden. Die Log-Likelihood bewertet die Punktwolken ausschließlich auf der Ebene, der geschätzten Anzahl von Linien, der Anzahl der Punkte pro Linie, sowie der Anzahl der Clutter-Punkte. Die räumliche Verteilung der Punkte innerhalb dieser Strukturen wird nicht modelliert. Wie in Algorithmus 3 beschrieben, werden die Punkte entlang der Linien gleichverteilt erzeugt. Für diese gleichverteilte Punktstruktur lässt sich daher keine sinnvolle Likelihood berechnen. Auch die Clutter-Punkte werden gleichverteilt und weisen daher auch keine sinnvolle Likelihood auf. Dadurch beschränkt sich die Likelihood-Bewertung auf den genannten Zählvariablen.

3.3. Methodik zur Evaluierung der FRD

Die Evaluierung der in Abschnitt 3.1 definierten FRD erfolgt in mehreren aufeinander aufbauenden Experimenten. Ziel ist es, das Distanzmaß in kontrollierten, synthetischen Szenarien zu validieren. Die Experimente sind so konzipiert, dass sie die in der Einleitung formulierte Forschungsfrage **RQ1 (Validität der Metrik)** systematisch adressieren.

Algorithmus 4 Heuristische Liniendetektion mit Clutter

Require: Punktmenge $\mathcal{P} \subset \mathbb{R}^d$, Schwellenwert $\tau > 0$, Mindestanzahl an Punkten pro Linie $N_{\min}$

lines $= [\,]$, clutter $= [\,]$
while $\mathcal{P} \neq \emptyset$ **do**
  if $|\mathcal{P}| < N_{\min}$ **then**
    clutter $=$ clutter $\cup\, \mathcal{P}$
    break
  point $=$ erstes Element aus $\mathcal{P}$
  lineFound $=$ **false**
  for point_line $\in \mathcal{P} \setminus \{\text{point}\}$ **do**
    Definiere Gerade L durch point und point_line
    $\mathcal{I} = \{\, p \in \mathcal{P} \mid \mathrm{dist}(p, L) \leq \tau \,\}$
    if $|\mathcal{I}| \geq N_{\min}$ **then**
      lines $=$ lines $\cup\, \{\mathcal{I}\}$
      $\mathcal{P} = \mathcal{P} \setminus \mathcal{I}$
      lineFound $=$ **true**
      break
  if *lineFound* = **false then**
    clutter $=$ clutter $\cup\, \{\text{point}\}$
    $\mathcal{P} = \mathcal{P} \setminus \{\text{point}\}$
return lines, clutter

Im ersten Schritt wird eine grundlegende Konsistenzeigenschaft der FRD überprüft. Dazu werden jeweils zwei unabhängige Stichproben aus demselben parametrischen Pseudoradar-Generator erzeugt. Erwartet wird, dass die FRD mit wachsender Stichprobengröße gegen Null konvergiert. Dieses Experiment dient der Validierung der Metrik in kontrollierten Szenarien und adressiert damit unmittelbar **RQ1 (Validität der Metrik)**.

Im zweiten Schritt wird die Trennfähigkeit der FRD untersucht. Hierzu werden Pseudoradar-Generatoren mit systematisch variierten Parametern eingesetzt, um unterschiedliche, aber strukturell verwandte Datenverteilungen zu erzeugen. In diesem Fall wird erwartet, dass die FRD nicht gegen Null konvergiert, sondern einen positiven Grenzwert annimmt, dessen Höhe vom Ausmaß der Parameterabweichung abhängt. Zur Einordnung der Ergebnisse, wird die approximierte Log-Likelihood unter dem jeweiligen Generator, als Referenzmaß herangezogen. Der Vergleich beider Größen erlaubt es, die Fähigkeit der FRD zu bewerten, Unterschiede zwischen Radar-Datenverteilungen konsistent abzubilden und liefert damit eine weitere empirische Evidenz für **RQ1**.

3.4. Methodik zur Evaluierung von Diffusionsmodellen zur Radar-Daten-Generierung

Nach der Evaluierung der FRD wird untersucht, ob Diffusionsmodelle für die Generierung synthetischer Radar-Daten geeignet sind. Der Schwerpunkt liegt dabei auf der generativen Eignung und dem Trainingsverhalten U-Net-basierter Diffusionsmodelle auf Radar-Daten (**RQ2**, **RQ4**). Die FRD wird in diesem Abschnitt nicht erneut als Metrik validiert, sondern als bereits abgesichertes Qualitätsmaß zur quantitativen Analyse der generierten Samples eingesetzt. Die Frage, ob die FRD die Entwicklung der Sample-Qualität während des Trainings widerspiegelt, wird ergänzend betrachtet (**RQ3**).

Als Modellarchitektur wird ein U-Net-basiertes Diffusionsmodell verwendet. Diese Architektur ist für Bilddaten entwickelt worden und eignet sich daher nicht für die Verarbeitung unstrukturierter Radar-Punktwolken [129]. Die in Abschnitt 3.1.1 eingeführte Diskretisierung überführt Radar-Punktwolken jedoch verlustfrei in eine bildähnliche, rasterbasierte Darstellung. Dadurch wird eine Eingabeform geschaffen, die mit U-Net-Architekturen kompatibel ist und ein Training auf Radar-Daten ermöglicht.

Die generative Eignung der Diffusionsmodelle wird anhand der Qualität der erzeugten Samples beurteilt. Dazu werden während des Trainings wiederholt Stichproben aus dem aktuellen Modell gezogen und mit Referenzdaten verglichen. Die quantitative Bewertung erfolgt über die FRD_{∞}, die als robuste, Stichproben-Bias-korrigierte Variante der FRD verwendet wird (vgl. Abschnitt 3.1.3.1). Damit lassen sich sowohl das erreichbare Qualitätsniveau als auch Veränderungen der Sample-Qualität über den Trainingsverlauf hinweg erfassen. Ergänzend wird die approximierte Log-Likelihood ausgewertet und zur Einordnung der FRD-Verläufe herangezogen. Qualitative Visualisierungen diskretisierter und rekonstruierter Punktwolken dienen der Plausibilisierung der Ergebnisse.

Zur Analyse des Einflusses der Modellkapazität werden mehrere U-Net-Varianten unterschiedlicher Größe betrachtet. Verglichen werden die Konvergenzgeschwindigkeit sowie das erreichbare Endniveau der FRD. Dieser Vergleich adressiert **RQ4** und erlaubt Aussagen darüber, wie stark die Modellgröße die Generierbarkeit und das Trainingsverhalten auf Radar-Daten beeinflusst.

4. Experimente

Im ersten Teil dieses Kapitels wird die in Kapitel 3 vorgestellte Fréchet-Radar-Distance (FRD) untersucht. Ziel der Experimente ist es, das Verhalten der FRD zu analysieren. Die Analyse erfolgt mithilfe von synthetischen Radar-Daten, die durch Pseudoradar-Generatoren generiert werden (siehe Abschnitt 3.2). Für die Analyse der FRD wird das asymptotische Verhalten in kontrollierten Situationen betrachtet. Dabei steht die Frage im Mittelpunkt, ob das Distanzmaß für identische Datenverteilungen mit wachsender Stichprobengröße konsistent gegen Null konvergiert und wie stark dieses Verhalten von der Projektionsdimension beeinflusst wird. Anschließend wird untersucht, wie sensibel die FRD auf gezielte Veränderungen der Datenverteilungen reagiert. Hierzu werden parametrische Pseudoradar-Generatoren mit variierenden Konfigurationen eingesetzt und die resultierenden Distanzwerte mit einer modellbasierten Referenzgröße verglichen. Zusammen zeigen diese Experimente, dass die FRD in synthetischen, parametrisch kontrollierten Radar-Szenarien konsistente, reproduzierbare und erwartungskonforme Bewertungen der Datenqualität liefert (**RQ1**).

Aufbauend auf der Validierung der FRD wird im zweiten Teil des Kapitels untersucht, ob Diffusionsmodelle für Radar-Daten geeignet sind. Im Mittelpunkt steht dabei die Frage, ob ein U-Net-basiertes Diffusionsmodell die durch den Pseudoradar-Generator generierte Datenverteilung lernen und realistische Samples erzeugen kann (**RQ2**). Ergänzend wird analysiert, wie sich die wahrgenommene Sample-Qualität während des Trainings verändert und ob diese Entwicklung durch die FRD konsistent abgebildet wird (**RQ3**). Abschließend wird der Einfluss der Modellkapazität auf die Sample-Qualität untersucht. Dafür werden mehrere U-Net-Varianten mit unterschiedlicher Kapazität hinsichtlich ihrer Konvergenzgeschwindigkeit und ihres erreichbaren Qualitätsniveaus verglichen (**RQ4**).

4.1. Evaluation der FRD in kontrollierten Szenarien

4.1.1. Konsistenz der FRD bei identischen Verteilungen

Bevor die in Abschnitt 3 vorgestellte FRD in komplexen Szenarien eingesetzt werden kann, wird in diesem Experiment zunächst eine notwendige Konsistenzeigenschaft untersucht. Konkret wird überprüft, ob die FRD für identische Verteilungen mit wachsender Stichprobengröße gegen Null konvergiert.

Hierzu wird ein Experiment durchgeführt, bei dem ein Pseudoradar-Daten-Generator gemäß Abschnitt 3.2 verwendet wird, um radarähnliche Punktwolken zu erzeugen. Es werden jeweils zwei unabhängige Datensätze generiert, deren Größe schrittweise erhöht wird. Für jedes Datensatzpaar gleicher Größe wird anschließend die in Abschnitt 3.1 beschriebene FRD berechnet.

Da die FRD auf der FD basiert und somit auf einem Vergleich von Mittelwert- und Kovarianzstrukturen beruht (siehe Gleichung 2.38), lautet die Hypothese dieses Experiments, dass die FRD für Paare von Datensätzen aus der gleichen Verteilung mit wachsender Anzahl an Datenpunkten gegen Null konvergiert [37]. Damit würde empirisch bestätigt, dass die Metrik für identische Verteilungen den theoretisch erwarteten Grenzwert annimmt. Die Umkehrung dieser Aussage gilt jedoch nicht, da eine FRD von Null im Allgemeinen keine vollständige Übereinstimmung der Verteilungen impliziert, wie für den allgemeinen Fall in Abschnitt 2.4.2 gezeigt.

4.1.1.1. Experimentaufbau

Der experimentelle Aufbau ist wie folgt. Zuerst werden Zufallsprojektionsmatrizen unterschiedlicher Zielraumdimensionen erzeugt. Ein Pseudoradar-Daten-Generator mit den Parametern:

$$\lambda_{\text{Linien}} = 5, \lambda_{\text{Punkte/Linie}} = 20, \lambda_{\text{Clutter}} = 50$$

erzeugt zwei Streams aus radarähnlichen Punktwolken (vgl. Abbildung 3.3). Die erzeugten Punktwolken werden bereits während der Generierung auf Gitter der Größe 64×64 diskretisiert, um zwei Streams diskretisierter Radar-Punktwolken zu erhalten. Die beiden Streams werden mithilfe der zuvor erzeugten Zufallsprojektionsmatrizen in niedrigdimensionaleren Räume unterschiedlicher Größe transformiert. In diesen Räumen werden die Mittelwerte und Kovarianzmatrizen der projizierten Daten mithilfe des Welford-Algorithmus berechnet. Nachdem eine definierte Anzahl von Stichproben durchlaufen worden ist, wird die FRD gemäß Gleichung 2.38 aus den gesammelten statistischen Größen auf den durch die zufälligen Projektionen niedrigdimensionaleren Räumen bestimmt.

Durch diesen Ablauf erhält man eine Auswertung der FRD in Abhängigkeit der Stichprobengröße. Dadurch lässt sich das Konvergenzverhalten in Bezug auf die Datensatzgröße sowie die FRD_∞ bestimmen. Zusätzlich lässt sich durch das Experiment der Einfluss der Projektionsdimension der zufälligen Projektion bestimmen. Hierfür wird die FRD für Paare an Stichproben mithilfe von Zufallsmatrizen unterschiedlicher Größe ausgewertet.

4.1.1.2. Ergebnisse

Zur Analyse des Konvergenzverhalten für die FRD-Werte aufsteigender Stichprobengröße wird FRD_∞ und das entsprechend an die Daten angepasste Re-

gressionsmodell herangezogen (siehe Abschnitt 3.1.3.1). Die in Abbildung 4.1 logarithmisch skalierten Ergebnisse zeigen das erwartete Konvergenzverhalten der FRD für identische Verteilungen. Es lässt sich erkennen, dass die Werte der FRD mit zunehmender Stichprobengröße gegen Null konvergieren. Die gestrichelten Linien repräsentieren die angepassten Regressionsmodelle, die zur Extrapolation auf $N \to \infty$ verwendet werden. Die logarithmisch skalierten Ergebnisse zeigen, dass die FRD mit zusätzlichen Samples exponentiell abfällt. Daraus folgt eine beschleunigte Konvergenz der FRD bei größeren Stichprobengrößen. Zusätzlich zu dem Plot sind die Ergebnisse des Regressionsmodells zur Schätzung der FRD_∞ in Tabelle 4.1 aufgeführt. Für alle Projektionsdimensionen liegt Null im Bereich des 95%-Konfidenzintervalls. Zusätzlich sind die p-Werte hoch, was darauf hinweist, dass die Hypothese $\text{FRD}_\infty = 0$ nicht verworfen werden kann. Insgesamt bestätigen die Ergebnisse sowohl die Konsistenzeigenschaft der FRD als auch die Notwendigkeit der Extrapolation zur Bestimmung von FRD_∞, um eine unverzerrte Schätzung zu erhalten. Die Ergebnisse reproduzieren damit die Ergebnisse von Neuhöfer und Reichardt [37]. Zusätzlich zu den hier dargestellten Ergebnissen werden unter Abschnitt B.1 die Ergebnisse zusätzlicher Experimente mit anderen Pseudoradar-Generatoren dargestellt.

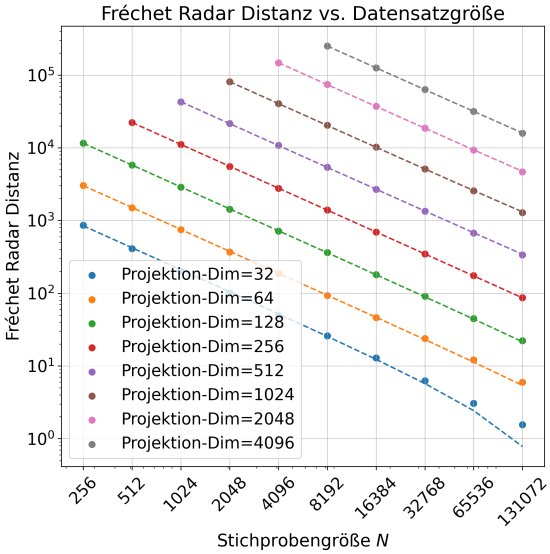

Abbildung 4.1.: FRD-Werte auf einer logarithmischen Skala für zwei Datensätze, die mit identischen Pseudoradar-Generator-Parametern erzeugt wurden. Die Ergebnisse sind für unterschiedliche Dimensionen der Zufallsprojektionsmatrizen dargestellt. Die angepassten Regressionsmodelle zur Extrapolation sind als gestrichelte Linien visualisiert.

Die Konvergenz der FRD gegen Null für Datensätze aus derselben Verteilung lässt sich auch theoretisch begründen. Die Berechnung der FRD basiert auf der Ähnlichkeit der Mittelwerte und der Kovarianzmatrix zweier Stichproben (siehe Abschnitt 2.4.2). Werden die Daten beider Stichproben durch denselben Pseudoradar-

Tabelle 4.1.: Regressionsergebnisse zur Bestimmung von FRD_∞ für verschiedene Projektionsdimensionen.

Dimension	**FRD_∞**	**95%-CI für FRD_∞**	**p-Wert**
32	0.17448	[-0.72432, 1.0733]	0.67
64	0.34602	[-1.2889, 1.9809]	0.64
128	4.1394	[-7.9676, 16.246]	0.45
256	7.4738	[-6.8788, 21.826]	0.26
512	18.428	[-19.025, 55.882]	0.27

Generator erzeugt, besitzen sie im ursprünglichen Datenraum identische Mittelwerte und dieselbe Kovarianzmatrix. Für beide Stichproben gilt, dass sie durch die zufällige Projektion mit derselben linearen Transformation transformiert werden. Mathematisch entspricht dies einer Betrachtung der Daten entlang der durch die Zufallsmatrix vorgegeben Richtungen. Werden nun im Bildraum der zufälligen Projektion die Kovarianzmatrizen der beiden Stichproben bestimmt, entspricht dies einer Projektion der ursprünglichen Kovarianzmatrix entlang der vorgegebenen Richtungen. Diese Eigenschaft folgt aus der Linearität der Kovarianzmatrix für einen Zufallsvektor $X \in \mathbb{R}^d$, seine Kovarianzmatrix $\mathrm{cov}(X)$ und der Zufallsmatrix $R \in \mathbb{R}^{k \times d}$ mit $k < d$ gilt:

$$\begin{aligned} \mathrm{cov}(RX) &= \mathbb{E}\big[(RX - R\mathbb{E}[X])(RX - R\mathbb{E}[X])^T\big] \\ &= \mathbb{E}\big[R(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T R^T\big] \\ &= R\,\mathbb{E}\big[(X - \mathbb{E}[X])(X - \mathbb{E}[X])^T\big]\, R^T \\ &= R\ \mathrm{cov}(X)\, R^T. \end{aligned} \tag{4.1}$$

Für die Mittelwerte $\mathbb{E}[X]$ gilt:

$$\mathbb{E}[RX] = \mathbb{E}[RX] = R\,\mathbb{E}[X]. \tag{4.2}$$

Da die ursprüngliche Kovarianzmatrix und die Mittelwerte der beiden Stichproben identisch sind, sind sie entlang der betrachteten Richtungen unter idealisierten Bedingungen identisch. Mit idealisierten Bedingungen ist eine unendliche Stichprobe gemeint. Denn unter der Annahme einer unendlichen Stichprobe verschwinden die Schätzfehler für Mittelwert und Kovarianzmatrix vollständig. In der Praxis bleibt jedoch ein Restfehler aufgrund endlicher Stichproben und numerischer Approximationen bestehen, sodass die FRD nur gegen Null konvergiert, aber nicht Null erreicht.

Zusätzlich zur Erkenntnis, dass die FRD für gleiche Radar-Daten mit wachsender Stichprobengröße gegen Null konvergiert, zeigt sich durch empirische Untersuchung, dass sich das Konvergenzverhalten als Funktion der Projektionsdimension und der Stichprobengröße darstellen lässt. Es lässt sich ein Skalierungsgesetz der

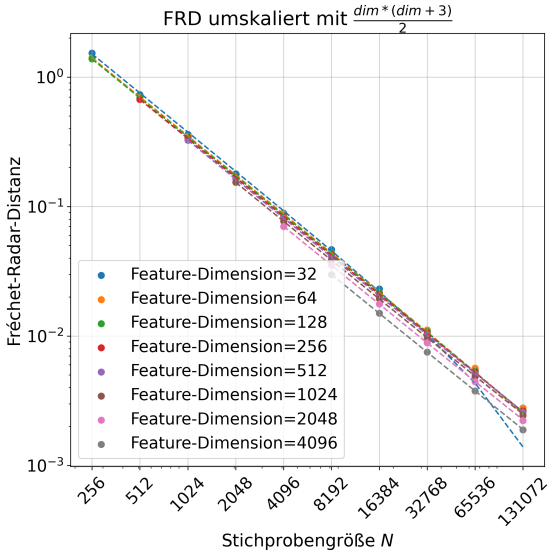

Abbildung 4.2.: Skalierte FRD-Werte für zwei Datensätze, die mit identischen Pseudoradar-Generator-Parametern erzeugt wurden. Die Ergebnisse sind für unterschiedliche Dimensionen der Zufallsprojektionsmatrizen dargestellt und wurden durch Division mit der effektiven Anzahl der zu schätzenden Parameter normalisiert um den Einfluss der Projektionsdimension zu entfernen.

Form:

$$\mathrm{FRD}(n,k) \;\sim\; \mathrm{FRD}_\infty \cdot \frac{k(k+3)}{2} \;+\; b\,\frac{1}{n} \cdot \frac{k(k+3)}{2}, \tag{4.3}$$

finden. Dieses Verhalten ist in Abbildung 4.2 dargestellt. Dort ist zu erkennen, dass die FRD-Verläufe für unterschiedliche Projektionsdimensionen nach der Umskalierung nahezu identisch verlaufen. Die beobachtete Dimensionsabhängigkeit lässt sich nicht durch Eigenschaften der Datenverteilungen erklären, da die Daten für alle Dimensionen identisch sind. Da die FRD unter idealisierten Bedingungen für Daten aus derselben Verteilung und für beliebige Zufallsprojektionsmatrizen asymptotisch gegen Null konvergiert, kann das beobachtete Verhalten nicht auf systematische Unterschiede der Dimensionalität der Daten zurückgeführt werden. Es ergibt sich jedoch eine theoretische Erklärung für dieses Verhalten. Sie leitet sich aus der Anzahl der Parameter, die für die Schätzung der Mittelwerte und Kovarianzmatrizen im projizierten Merkmalsraum erforderlich sind ab. Während die Anzahl der Mittelwerte mit der Projektionsdimension k linear wächst, besitzt die zugehörige Kovarianzmatrix der Dimension $k \times k$ insgesamt $\frac{k(k+1)}{2}$ freie Parameter. Zusammen mit den Mittelwerten ergibt sich somit eine effektive Anzahl von

$$\frac{k(k+3)}{2}$$

zu schätzenden Parametern. Mit zunehmender Projektionsdimension steigt dadurch die Komplexität der Schätzung, was insbesondere bei endlicher Stichpro-

bengröße zu einer erhöhten Varianz der geschätzten Mittelwerte und Kovarianzmatrizen führt. Die FRD basiert auf den Abweichungen der Mittelwerte und der Kovarianzmatrizen. Weisen die Schätzungen der statistischen Größen eine höhere Varianz auf, so erhöht sich auch die erwartete Abweichung, die die FD misst. Das führt dazu, dass die FRD für größere Projektionsdimensionen bei endlicher Stichprobengröße systematisch höhere Werte annimmt.

Insgesamt zeigen die Ergebnisse sowohl theoretisch als auch experimentell, dass die FRD für identische Verteilungen gegen Null konvergiert und die Projektionsdimension einen dominanten Einfluss auf die absoluten Werte des Distanzmaßes hat. Die Ergebnisse beantworten damit einen Teil von **RQ1**, der sich auf die Validität in kontrollierten Szenarien bezieht. Zusätzlich zur Konvergenz gegen Null für identische Verteilungen muss für die abschließende Antwort auf **RQ1** die Fähigkeit nachgewiesen werden, Unterschiede zwischen verschiedenen Radar-Datenverteilungen zuverlässig abzubilden.

4.1.2. Sensitivität der FRD gegenüber Verteilungsänderungen

Nachdem im vorherigen Experiment die Konsistenzeigenschaft der FRD für identische Verteilungen untersucht und gezeigt wurde, wird im Folgenden untersucht, wie gut die FRD Unterschiede zwischen verschiedenen Radar-Datenverteilungen abbildet. Dazu wird untersucht, wie sich die FRD verhält, wenn die Pseudoradar-Generatoren unterschiedliche Parameter besitzen und somit unterschiedliche Verteilungen generieren. Ziel ist es, empirisch zu zeigen, dass die FRD in diesem Fall nicht gegen Null konvergiert, sondern einen positiven Grenzwert annimmt, dessen Höhe vom Grad der Parameterabweichung abhängt.

Dafür wird ein Experiment durchgeführt, bei dem mehrere Pseudoradar-Daten-Generatoren gemäß Abschnitt 3.2 verwendet werden, um mit unterschiedlichen Parametern radarähnliche Punktwolken zu erzeugen. Wie im vorherigen Experiment werden zwei unabhängige Datensätze generiert, deren Größe schrittweise erhöht wird um die FRD zu berechnen.

Die Hypothese lautet, dass die FRD für nahezu identische Generatorparameter weiterhin sehr kleine Werte liefert, während sie für stark abweichende Konfigurationen deutlich höhere Werte annimmt. Als Referenzmaß wird die approximierte Log-Likelihood der Daten unter dem jeweiligen Generator herangezogen. Sie liefert ein theoretisches Sample-basiertes Distanzmaß, das die Ähnlichkeit zwischen den Daten aus verschiedenen Generatoren bewertbar macht (siehe Abschnitt 3.2.1). Durch den Vergleich der FRD mit der approximierten Log-Likelihood lässt sich die Fähigkeit der FRD bewerten, Unterschiede zwischen verschiedenen Radar-Datenverteilungen abzubilden. Mithilfe des Experiments wird untersucht, ob die FRD in der Lage ist, strukturelle Unterschiede in den erzeugten Radar-Punktwolken zu erkennen und diese konsistent mit der approximierten Log-Likelihood abzubilden.

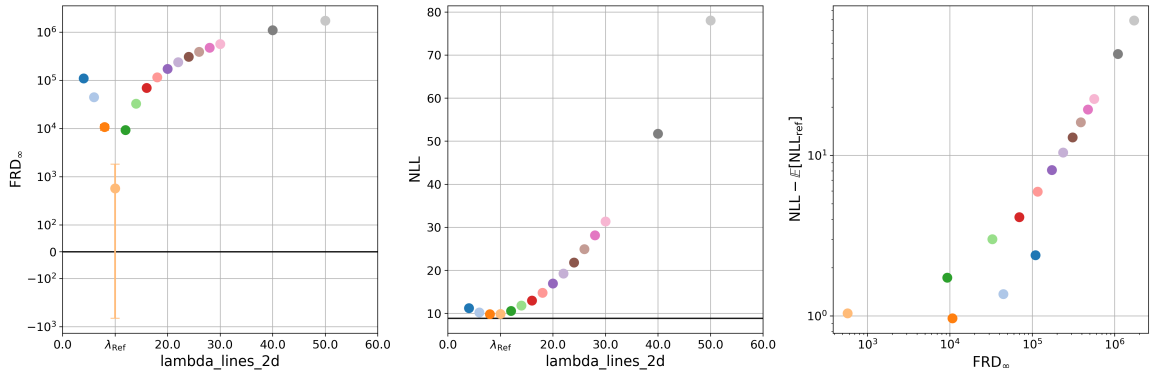

Abbildung 4.3.: Visualisierung der FRD_∞ für die Feature-Dimension 4096, der approximierten Log-Likelihood sowie ihres Zusammenhangs für Radar-Generatoren, bei denen ausschließlich die Anzahl der Linien variiert wird. Die Anzahl der Clutter-Punkte und der Punkte pro Linie ist für alle Generatoren konstant und entspricht den Referenzparametern $\lambda_{\mathrm{Punkte/Linie}} = 20$ und $\lambda_{\mathrm{Clutter}} = 50$. Der linke Plot zeigt die FRD_∞-Werte, der mittlere die approximierte Log-Likelihood und der rechte den Zusammenhang beider Größen, wobei die Log-Likelihood relativ zur erwarteten Log-Likelihood skaliert ist.

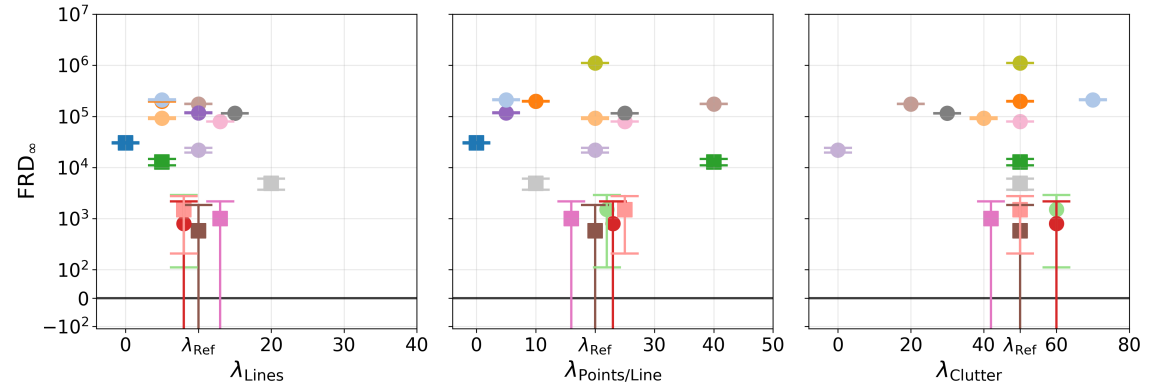

Abbildung 4.4.: Visualisierung der FRD_∞ verschiedener Radar-Generatoren bei einer Feature-Dimension von 4096. Die drei Subplots zeigen jeweils einen der Generierungsparameter, wobei jeder Generator über alle Subplots hinweg konsistent farbkodiert ist. Jede Markierung repräsentiert die FRD_∞ eines Generators relativ zum Referenzgenerator, dessen Parameter mit λ_{Ref} gekennzeichnet sind. Generatoren mit identischer erwarteter Punktanzahl (vgl. Gleichung 3.6) sind als Quadrate dargestellt, alle Übrigen als Kreise. Vertikale Linien geben das 95%-Konfidenzintervall der FRD_∞-Schätzung an.

4.1.2.1. Experimentaufbau

Der Versuchsaufbau entspricht grundsätzlich dem in Abschnitt 4.1.1 beschriebenen Experiment mit identischen Generatorparametern. Der Unterschied besteht darin, dass ein Referenzdatensatz mit den Parametern:

$$\lambda_{\text{Linien}} = 10, \lambda_{\text{Punkte/Linie}} = 20, \lambda_{\text{Clutter}} = 50$$

generiert wird, sowie mehrere Vergleichsdatensätze, die mit unterschiedlichen Parametern generiert werden. Die Parameter der Vergleichsgeneratoren variieren dabei systematisch in mindestens einem Parameter, um unterschiedliche Ähnlichkeitsgrade zu simulieren. Für jeden Vergleichsdatensatz wird die FRD zum Referenzdatensatz mit aufsteigender Anzahl an Daten berechnet.

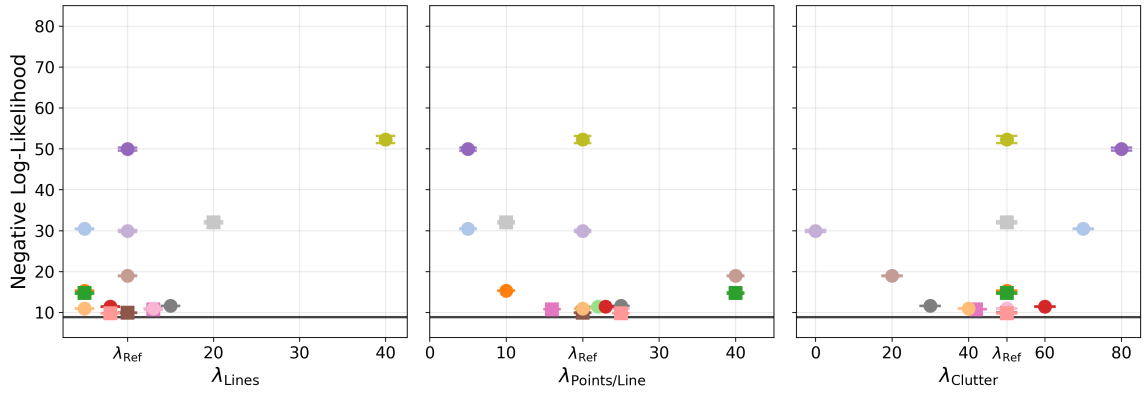

Abbildung 4.5.: Visualisierung der geschätzten Log-Likelihood verschiedener Radar-Generatoren. Die drei Subplots zeigen jeweils einen Generierungsparameter, wobei jeder Generator über alle Subplots hinweg konsistent farbkodiert ist. Jede Markierung entspricht der Log-Likelihood eines Generators relativ zu den Referenzparametern λ_{Ref}. Generatoren mit identischer erwarteter Punktanzahl (vgl. Gleichung 3.6) sind als Quadrate dargestellt, alle anderen als Kreise.

4.1.2.2. Ergebnisse

Auch in diesem Experiment wird das Konvergenzverhalten anhand der FRD_∞ analysiert. Zunächst werden Generatoren betrachtet, die sich nur in einem der drei Parameter vom Referenzgenerator unterscheiden. Die Ergebnisse sind in Abbildung 4.3 und in Abschnitt B.2 dargestellt. In den Abbildungen kann man erkennen, dass sowohl die FRD_∞ als auch die approximierte Log-Likelihood mit zunehmender Abweichung des jeweiligen Parameters vom Referenzgenerator

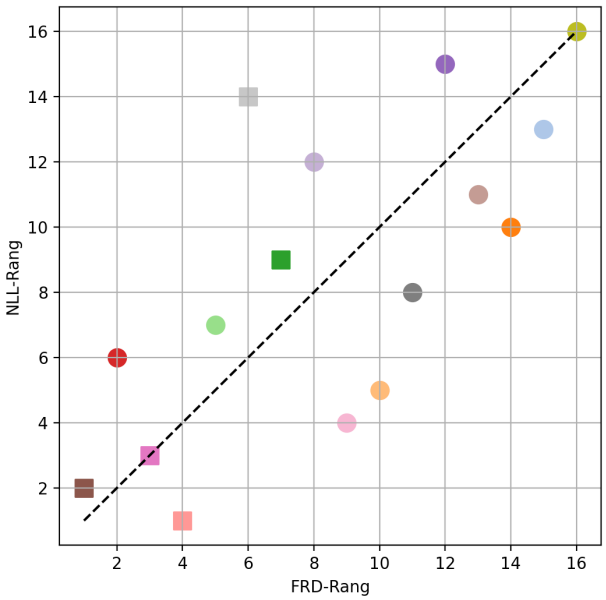

Abbildung 4.6.: Rangkorrelationen zwischen der FRD_∞ und der approximierten Log-Likelihood für verschiedene Radar-Generatoren darstellt. Die FRD wurde mit der Feature-Dimension 4096 ausgewertet.

ansteigt. Zusätzlich wird in der dritten Spalte die Abhängigkeit der beiden Größen dargestellt. Dabei wird die Log-Likelihood mit der erwarteten Log-Likelihood skaliert.

Es lässt sich sowohl für die FRD_∞ als auch für die Log-Likelihood erkennen, dass sie nicht symmetrisch auf Änderungen in positiver und negativer Richtung reagieren. Zur Einordnung dieses Experiments ist jedoch einzuräumen, dass durch die Veränderung nur eines Parameters die Anzahl der erwarteten Punkte variiert, was in Bezug auf die Log-Likelihood zwar irrelevant ist, aber einen Einfluss auf die FRD hat. Das liegt daran, dass die Anzahl der Punkte direkt die Kovarianzmatrix im Originalraum beeinflusst, was sich wiederum direkt auf die Kovarianzmatrix im Projektionsraum und somit auf die FRD auswirkt.

Aus diesem Grund werden im folgenden Experiment Generatoren betrachtet, die sich in mehreren Parametern unterscheiden. Die Ergebnisse werden in den folgenden Abbildungen dargestellt. Über alle Abbildungen sind Generatoren mit einer Farbe kodiert, die konsistent für dieselben Generatoren sind. Zusätzlich wird zwischen Generatoren mit derselben und unterschiedlichen Anzahl an erwarteten Punkten (siehe Abschnitt 3.2) unterschieden. Generatoren, die dieselbe Anzahl an erwarteten Punkten wie der Referenzgenerator erzeugen werden mit einem Rechteck dargestellt. Generatoren die mehr oder weniger erwartete Punkte erzeugen werden mit einem Kreis dargestellt. In Abbildung 4.4 kann man erkennen, dass Parameter Konfigurationen, die eine ähnliche Anzahl an erwarteten Punkten

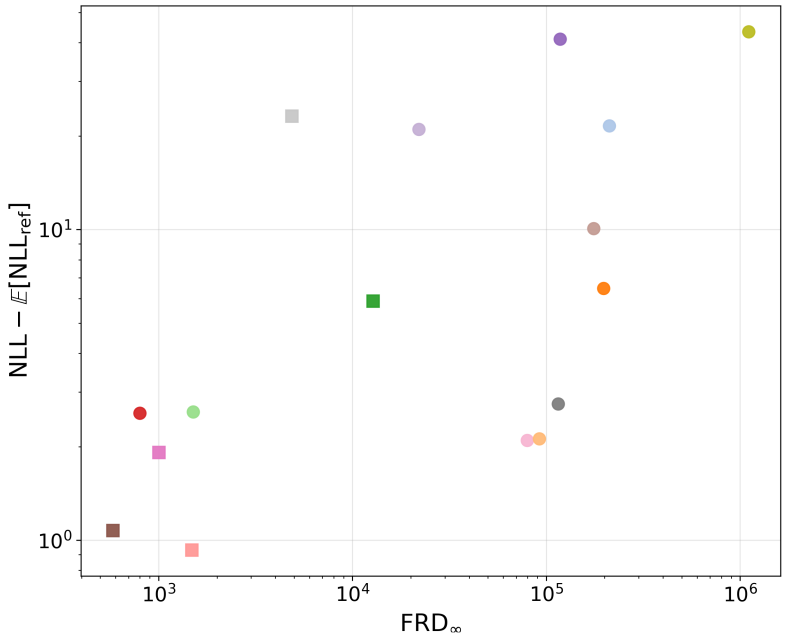

Abbildung 4.7.: Die Wechselwirkungen zwischen der FRD_∞ und der approximierten Log-Likelihood Werte für verschiedene Radar-Generatoren darstellt. Die FRD wurde mit der Feature-Dimension 4096 ausgewertet.

erzeugen, im Vergleich zu denen, die eine andere Anzahl an Punkten erzeugen, durchschnittlich niedrigere FRD_∞-Werte erreichen. Da sich die Ähnlichkeit der von verschiedenen Generatoren erzeugten Daten nicht eindeutig beurteilen lässt, wird in Abbildung 4.5 die approximierte Log-Likelihood der verschiedenen Generatoren als Vergleichsmaß dargestellt. Um die Ergebnisse zwischen der FRD_∞ und der Log-Likelihood besser zu vergleichen, wird in Abbildung 4.6 die Rangkorrelation zwischen den Rängen der FRD_∞ und der approximierten Log-Likelihood dargestellt. Es lässt sich eine Rangkorrelation zwischen der FRD_∞ und der approximierten Log-Likelihood erkennen. Die Korrelation der vergebenen Ränge sind in Tabelle 4.2 gegeben. Insgesamt zeigen alle betrachteten Rangkorrelationsmaße eine Korrelation zwischen den beiden Rangfolgen auf. Zusätzlich wurde die Übereinstimmung der Top-k-Generatoren in beiden Rankings analysiert. Dabei zeigt sich, dass unter den Top-3-Generatoren nur eine geringe Übereinstimmung besteht. Dies ist jedoch plausibel, da die erstplatzierten Generatoren in beiden Rankings sehr nahe beieinander liegen und kleine Abweichungen in der Bewertung zu Rangverschiebungen führen können. Für die Top-5- und Top-10-Generatoren ergibt sich hingegen eine deutlich höhere Übereinstimmung. Diese Ergebnisse deuten darauf hin, dass die FRD_∞ die Ähnlichkeit der Pseudoradar-Generatoren insgesamt konsistent zur approximierten Log-Likelihood bewertet.

Zusätzlich werden die konkreten Werte für die verschiedenen Generatoren bezüglich der FRD_∞ und der approximierten Log-Likelihood in Abbildung 4.7 dargestellt. Es lässt sich ein linearer Zusammenhang mit einzelnen Ausreißer erkennen.

Tabelle 4.2.: Rangkorrelationsmaße zwischen der FRD_{∞} und der approximierten Log-Likelihood für verschiedene Radar-Generatoren. P@k gibt an, wie viele der Top-k Generatoren in beiden Rankings übereinstimmen.

Kendall	Spearman	Pearson	Rang MAE	P@3	P@5	P@10
0.5333	0.6971	0.6376	3.0000	2/3	3/5	8/10

Aus den beiden Darstellungen lässt sich ein klarer, jedoch nicht perfekter Zusammenhang im Wertverhalten sowie im Ranking der beiden Größen erkennen. Die Abweichungen können sich dadurch erklären lassen, dass die beiden Metriken unterschiedliche Aspekte der Generatorähnlichkeit bewerten. Die approximierte Log-Likelihood misst die Abweichung der generierten Punktwolken im Sinne des Poisson-basierten generativen Modells und reagiert daher direkt auf Veränderungen der Modellparameter. Die FRD_{∞} hingegen misst Unterschiede in den ersten zwei Momenten der Verteilungen der generierten Punktwolken im durch Zufallsprojektion definierten Feature-Raum. Diese Momente werden stark von einer abweichenden Punktanzahl beeinflusst, da die Kovarianzmatrix im Originalraum und somit auch im projizierten Raum direkt von der Anzahl der Punkte abhängt. Zusätzlich müssen zur Einordnung der Ergebnisse die Schwächen der approximierten Log-Likelihood berücksichtigt werden, die in Abschnitt 3.2.1 beschrieben werden.

Trotz der Unterschiede zeigen beide Metriken eine Korrelation, was darauf hinweist, dass sie beide relevante Informationen über die Ähnlichkeit der Generatoren liefern. Die approximierte Log-Likelihood misst direkt die Abweichung der approximierten Modellparameter und stellt damit eine interpretierbare Bewertungsgröße dar. Da sie in dieser Arbeit als Referenzmetrik verwendet wird, ist die beobachtete Korrelation ein Hinweis darauf, dass die FRD_{∞} für die Bewertung von Punktwolken aus Pseudoradar-Generatoren geeignet ist. Zusätzliche Ergebnisse, die zu demselben Ergebnis kommen, sind in Abschnitt B.3 dargestellt. Insgesamt zeigen die Ergebnisse, dass die FRD ein geeignetes Maß ist, die Ähnlichkeit zwischen Generatoren zu messen. Damit beantworten die Experimente **RQ1**. Es wurde gezeigt, dass die FRD in kontrollierten Szenarien konsistente und erwartungskonforme Bewertungen liefert.

4.2. Diffusionsmodelle für Pseudoradar-Daten

4.2.1. Training eines U-Net-basierten Diffusionsmodells auf Pseudoradar-Daten

In diesem Experiment wird die generative Eignung eines U-Net-basierten Diffusionsmodells für Radar-Daten untersucht (**RQ2**). Als Datenbasis dienen Punktwolken aus den parametrisierten Pseudoradar-Generatoren (vgl. Abschnitt 3.2), die eine

kontrollierte und reproduzierbare Zielverteilung definieren. Das Diffusionsmodell wird auf den diskretisierten Punktwolken trainiert (vgl. Abschnitt 3.1.1). Zur Bewertung der generierten Samples werden in regelmäßigen Abständen Daten mit dem aktuellen Modell generiert und die FRD_∞ zwischen Modell-Samples und Referenzdaten berechnet. Die FRD dient dabei als primäres, quantitatives Qualitätsmaß.

Zusätzlich wird in diesem Experiment die Frage betrachtet, ob die FRD die qualitative Entwicklung der Samples während des Trainings konsistent abbildet (**RQ3**). Dafür wird die approximierte Log-Likelihood unter dem jeweiligen Pseudoradar-Generator als Referenzmaß verwendet (vgl. Abschnitt 3.2.1). Da die Likelihood-Berechnung auf einer heuristischen Liniendetektion basiert, werden Abweichungen von idealen Linien durch eine Distanz ϵ toleriert. ϵ ist dabei ein Hyperparameter und dient ausschließlich dazu, das Referenzmaß auf Samples anzuwenden, die nicht exakt dem Generator-Modell entsprechen. Ergänzend werden Stichproben visualisiert, um die Rekonstruktion der dominanten linearen Strukturen visuell zu überprüfen.

Die Hypothese des Experiments ist, dass das U-Net-basierte Diffusionsmodell mit fortschreitendem Training in der Lage ist, die durch den Pseudoradar-Generator definierte Zielverteilung zu approximieren und realistische Radar-Punktwolken zu erzeugen (**RQ2**). Diese Verbesserung der Sample-Qualität sollte sich in einer abnehmenden FRD_∞ zeigen. Darüber hinaus wird angenommen, dass die FRD die qualitative Entwicklung der generierten Samples während des Trainings abbildet (**RQ3**). Änderungen in der Struktur der generierten Punktwolken, insbesondere in der Ausprägung linearer Reflexionsmuster, sollten mit systematischen Veränderungen der FRD_∞ einhergehen. Der Vergleich mit der approximierten Log-Likelihood sollte einen ähnlichen Verlauf zeigen und darauf hinweisen, dass die FRD_∞ als Metrik zur Evaluierung des Trainingsfortschritts geeignet ist.

4.2.1.1. Experimentaufbau

Das Experiment basiert auf Daten, die durch einen parametrisierten Pseudoradar-Generator erzeugt werden (vgl. Abschnitt 3.2). Für alle Trainings- und Evaluationsschritte wird ein Generator mit den festen Parametern:

$$\lambda_{\text{Linien}} = 10, \quad \lambda_{\text{Punkte/Linie}} = 20, \quad \lambda_{\text{Clutter}} = 50,$$

verwendet. Die erzeugten Punktwolken werden nach dem Generieren in eine gitterbasierte Darstellung der Größe 64×64 mit 4 Kanälen überführt (vgl. Abschnitt 3.1.1).

Für das Training wird eine feste U-Net-basierte Diffusionsarchitektur verwendet. Die Architektur entspricht der Modellvariante **M3**, die vollständig (Kanaltiefen, ResNet-Blöcke, Attention-Module sowie Parameteranzahl) in Tabelle 4.3 dokumentiert ist. Das Modell wird mit $T = 1000$ Diffusionsschritten trainiert und verwendet eine Cosine-Schedule (siehe Abschnitt 2.3.1.4). Das Trainingsziel ist die

x_0-Vorhersage und als Rekonstruktionsverlust wird der MSE verwendet. Das Training erfolgt mit einer Batchgröße von 32. Die Trainingsdaten werden während des Trainings on-the-fly aus dem Generator Stream gezogen. Aus diesem Grund ist jeder Batch eine neue Realisierung des Generators und es werden keine Samples mehrfach verwendet. Die Batches sind damit identisch verteilt und unabhängig. Diese Annahme stellt eine Idealisierung gegenüber einem endlichen Datensatz dar, aber ist für das vorliegende Experiment sinnvoll, da in erster Linie gezeigt werden soll, dass U-Net-Diffusionsmodelle grundsätzlich in der Lage sind, Radar-Daten zu generieren. Die Optimierung erfolgt mit Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$), Gradient-Clipping (maximale Norm 1.0) und einer initialen Lernrate von $5 \cdot 10^{-5}$.

Die Modellqualität wird zu festgelegten Trainingszeitpunkten evaluiert. Hierzu wird das Modell nach:

$$5000,\ 10000,\ 20000,\ 40000,\ \ldots,\ 900000,$$

Trainingsiterationen jeweils als Checkpoint gespeichert, und es werden aus dem aktuellen Modell synthetische Samples generiert. Für die Stichprobenerzeugung wird ein DDIM-Sampler mit Zehn Sampling-Schritten verwendet (vgl. Abschnitt 2.3.1.2). Die Berechnung von FRD_∞ erfolgt analog zu den vorherigen Experimenten und Abschnitt 3.1.

Zusätzlich wird ein separater, unabhängiger Datenstrom aus demselben Pseudoradar-Generator erzeugt. Für mehrere Stichprobengrößen werden die zugehörigen Mittelwerte und Kovarianzmatrizen im Projektionsraum geschätzt. Für jede Projektionsdimension

$$d \in \{512, 1024, 2048, 4096\}$$

wird jeweils eine Zufallsmatrix als Feature-Extraktor verwendet, sodass die $\mathrm{FRD}(n)$ für verschiedene Projektionsdimensionen bestimmt werden kann. Anschließend wird aus den $\mathrm{FRD}(n)$-Werten die Grenzwertschätzung FRD_∞ mittels Extrapolation gemäß Abschnitt 3.1.3.1 berechnet. Die Log-Likelihood wird ergänzend zur FRD_∞ als Referenzgröße herangezogen (vgl. Abschnitt 3.2.1) und dient nicht als primäres Qualitätsmaß, sondern ausschließlich zur Einordnung der durch FRD_∞ beobachteten Trainingsverläufe.

4.2.1.2. Ergebnisse

Abbildung 4.8 zeigt den Verlauf der FRD_∞ in Abhängigkeit vom Trainingsfortschritt des U-Net-Diffusionsmodells für verschiedene Projektionsdimensionen. Unabhängig von der gewählten Projektionsdimension ist in der frühen Trainingsphase ein starker Abfall der FRD_∞ zu beobachten. Dieser Abfall ist erwartbar unter der Annahme, dass das Modell in den ersten Iterationen beginnt, die sparse Struktur der Daten zu erfassen. Dieses Verhalten wird durch die Abbildung 4.9 verdeutlicht. Die Abbildung zeigt die Verteilung eines Validitätsindikators für generierte

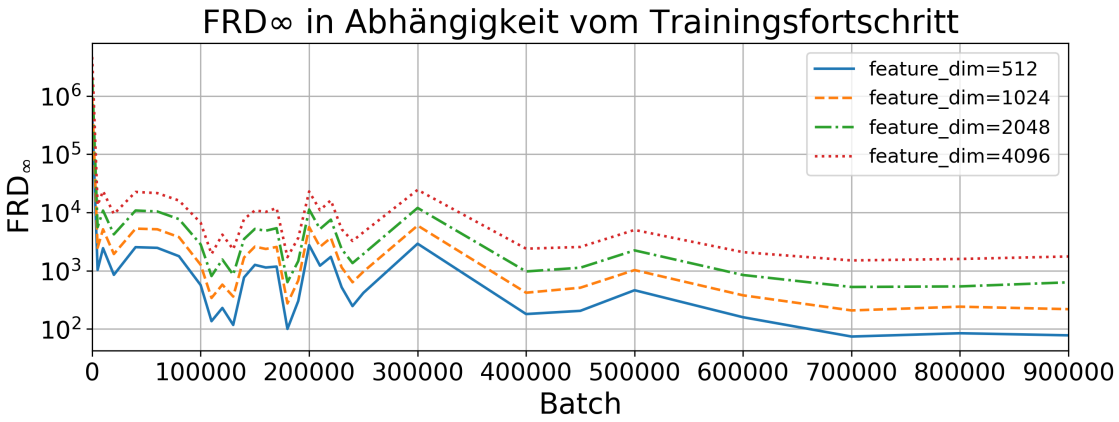

Abbildung 4.8.: Verlauf der FRD_∞ über den Trainingsfortschritt für verschiedene Projektionsdimensionen.

Samples vor dem Training sowie nach $5\,000$ Trainingsiterationen. Der Indikator beschreibt, ob eine Gitterzelle einen Punkt enthält. Da der Pseudoradar-Generator im Erwartungswert 250 Punkte erzeugt, besteht die Referenzverteilung aus 250 Einträgen mit dem Wert 1 sowie $64 \times 64 - 250$ Einträgen mit dem Wert 0. Während das Diffusionsmodell diese Struktur vor dem Training nicht abbildet, ist nach $5\,000$ Trainingsiterationen bereits eine deutlich bessere Annäherung erkennbar. Da die FRD_∞ sensibel auf Änderungen der Punktanzahl reagiert, spiegelt sich diese frühe Anpassung globaler Verteilungseigenschaften im starken initialen Abfall der Metrik wider.

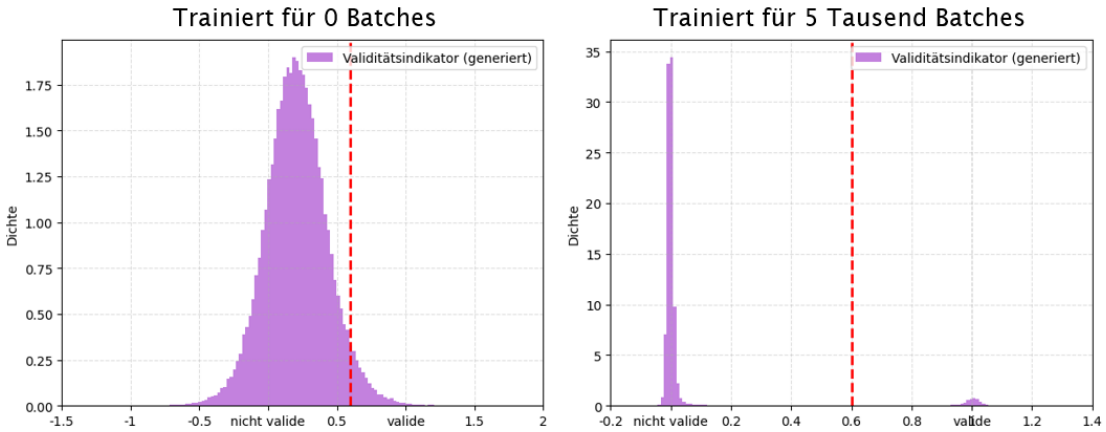

Abbildung 4.9.: Verteilung des Validitätsindikators für generierte Samples zu Beginn des Trainings (links) und nach $5\,000$ Trainingsiterationen (rechts). Die markierte Linie bei 0.6 kennzeichnet den Schwellenwert, ab dem eine Gitterzelle einen Punkt enthält.

Nach dem initialen Abfall der FRD_∞ folgt eine Phase erhöhter Varianz. In diesem Abschnitt zeigt die Metrik deutliche Schwankungen und kein monotones Kon-

vergenzverhalten. Das legt nahe, dass das Modell zwar bereits grundlegende globale Verteilungseigenschaften reproduziert, die generierten Samples jedoch noch nicht konsistent der Zielverteilung folgen. Dieses Verhalten zeigt sich auch in Abbildung 4.10, in der generierte Punktwolken zu verschiedenen Trainingszeitpunkten dargestellt sind. Nach $100\,000$ Trainingsiterationen lassen sich bereits lineare Strukturen erkennen, die jedoch noch nicht vollständig mit den Referenzdaten übereinstimmen. Insbesondere treten lokale Abweichungen von idealer Linearität auf. Erst in späteren Trainingsphasen nehmen diese Abweichungen ab, und die generierten Punktwolken nähern sich zunehmend den in den Trainingsdaten enthaltenen Strukturen an. Mit dieser Entwicklung geht eine Stabilisierung der FRD_∞ einher. In diesem Bereich nimmt die Varianz der Metrik deutlich ab, und die FRD_∞ konvergiert auf ein nahezu konstantes Niveau. Dies deutet darauf hin, dass das Modell die Zielverteilung nicht mehr nur grob approximiert, sondern konsistent reproduziert.

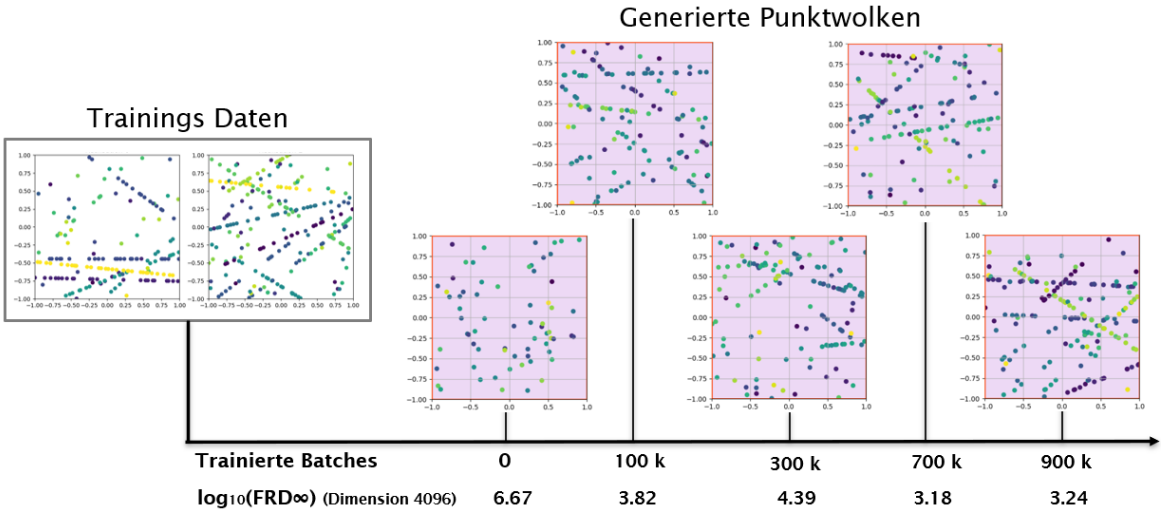

Abbildung 4.10.: Generierte Radar-Punktwolken zu ausgewählten Trainingszeitpunkten. Dargestellt sind die aus dem generierten Gitter rekonstruierten Punkte mit Validitätsscore > 0.6. Links sind Trainingsdaten als Referenz gezeigt. $\log_{10}(\mathrm{FRD}_\infty)$ ($d = 4096$) gibt die quantitative Modellqualität an.

In Abbildung 4.8 erkennt man, dass die FRD_∞ für höhere Projektionsdimensionen größere absolute Werte annimmt. Dieses Verhalten hat sich bereits in den vorherigen Experimenten gezeigt und wurde in Abschnitt 4.1.1.2 theoretisch begründet. Der qualitative Verlauf über den Trainingsfortschritt sowie die relative Rangordnung der Modellzustände bleiben über alle betrachteten Dimensionen hinweg konsistent. Dieses Verhalten ist zunächst überraschend. Es liegt nahe, dass sich die FRD-Werte für unterschiedliche Projektionsdimensionen unterscheiden, da die Projektionsdimension bestimmt, in welchem Maß strukturelle Unterschiede der Punktwolken im Projektionsraum erhalten bleiben (vgl. Abschnitt 2.5).

Zur Einordnung der im Training erreichten FRD_∞-Werte werden diese mit den Ergebnissen aus dem Experiment zu Pseudoradar-Generatoren mit variierenden

Parametern verglichen (vgl. Abschnitt 4.1.2.2). Abbildung 4.11 stellt die FRD_∞ des trainierten Diffusionsmodells mit den FRD_∞-Werten der Pseudoradar-Generatoren gegenüber. Die im Training erreichten FRD_∞-Werte sind im Bereich der Werte der sechs ähnlichsten Pseudoradar-Generatoren aus dem vorherigen Experiment. Das bedeutet nicht, dass die vom Diffusionsmodell erzeugten Punktwolken einer konkreten Generatorparametrisierung entsprechen. Der Vergleich dient nur dazu, die Größenordnung der FRD_∞ einzuordnen und soll keine Gleichsetzung des Diffusionsmodells und einer konkreten Generatorparametrisierung suggerieren.

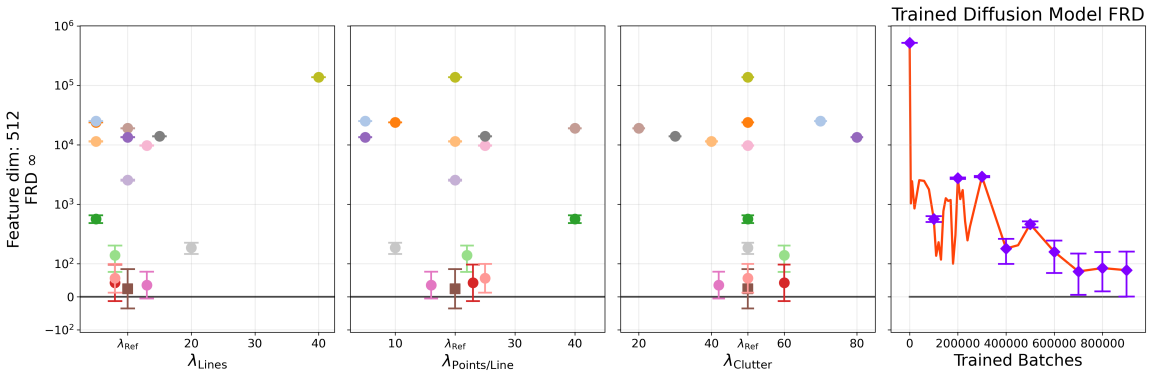

Abbildung 4.11.: FRD_∞ (Feature-Dimension $d = 512$) für variierte Pseudoradar-Generator-Parameter (links) und für das trainierte Diffusionsmodell über den Trainingsfortschritt (rechts).

Zusätzlich zur Analyse der FRD_∞ und der generierten Samples ist in Abbildung 4.12 der Verlauf der negativen Log-Likelihood dargestellt. Auch hier ist zu Beginn des Trainings ein starker Abfall zu beobachten. Nach dieser frühen Phase geht die negative Log-Likelihood in ein stabiles Plateau über und verändert sich im weiteren Trainingsverlauf nur noch geringfügig. Im Vergleich dazu stabilisiert sich die FRD_∞ erst deutlich später. Dieser Unterschied lässt sich unter anderem durch die Wahl des Hyperparameters der Liniendetektion der approximierten Log-Likelihood erklären. Er bestimmt, wie nah Punkte an einer Linie sein müssen, um ihr zugeordnet zu werden. Um eine stabile Liniendetektion zu gewährleisten, wurde hierfür ein Schwellenwert von 0.10 gewählt. Die Log-Likelihood ist dadurch weniger sensitiv gegenüber feinen geometrischen Abweichungen. Die Abweichungen von ideal linearen Strukturen, wie sie in Abbildung 4.10 zu erkennen sind, werden somit von der approximierten Log-Likelihood nicht erfasst.

Zusammenfassend zeigen die Ergebnisse, dass das U-Net-basierte Diffusionsmodell in der Lage ist, die durch den Pseudoradar-Generator definierte Zielverteilung konsistent zu approximieren (**RQ2**). Dies zeigt sich sowohl im Verlauf und in der Konvergenz der FRD_∞ als auch in der visuellen Analyse der generierten Punktwolken.

Die Ergebnisse sprechen zudem dafür, dass die FRD die qualitative Entwicklung der generierten Samples während des Trainings konsistent abbildet (**RQ3**). Veränderungen in der globalen Struktur der Punktwolken und die zunehmende

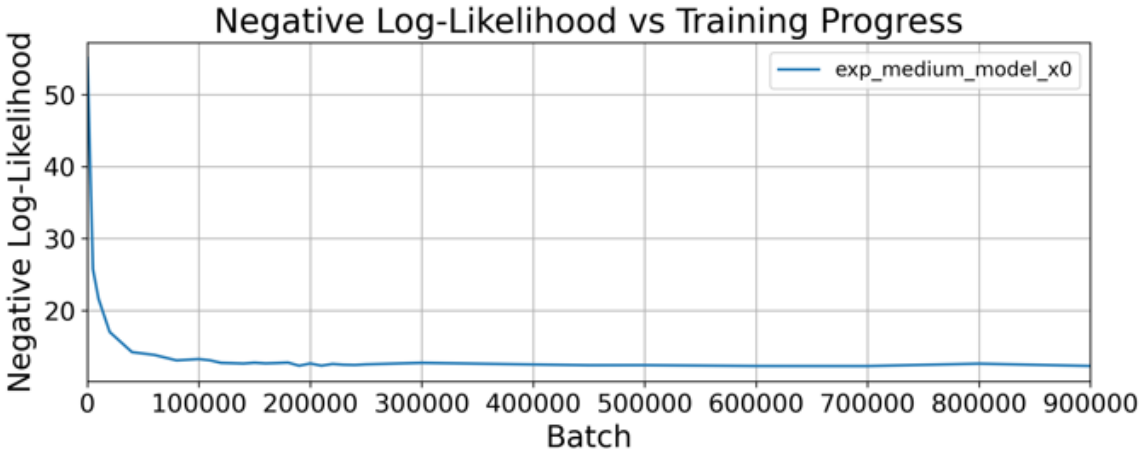

Abbildung 4.12.: Verlauf der negativen Log-Likelihood in Abhängigkeit vom Trainingsfortschritt.

geometrische Konsistenz der Linienmuster gehen mit systematischen Veränderungen der FRD_∞ einher. Der Vergleich mit der approximierten Log-Likelihood zeigt ähnliche frühe Trainingsphasen, jedoch Unterschiede im Stabilitätsverhalten. Damit eignet sich die FRD_∞ zur Bewertung des Trainingsfortschritts. Sie erfasst globale Verteilungseigenschaften und spiegelt strukturelle Änderungen der generierten Samples wider.

4.2.2. Einfluss der Modellkapazität auf die Generierung von Radar-Daten

Dieses Experiment untersucht den Einfluss der Modellkapazität auf das Trainingsverhalten und die erreichbare Sample-Qualität eines U-Net-basierten Diffusionsmodells (**RQ4**). Dazu werden mehrere Modellvarianten mit unterschiedlicher Kapazität unter identischen Trainingsbedingungen verglichen. Als Bewertungsmaß wird die FRD_∞ verwendet, sodass sowohl die Konvergenzgeschwindigkeit als auch das erreichbare Endniveau der Sample-Qualität quantifiziert werden kann. Qualitative Visualisierungen dienen zur Plausibilisierung der Unterschiede zwischen den Modellvarianten.

4.2.2.1. Experimentaufbau

Der experimentelle Aufbau entspricht weitgehend dem in Abschnitt 4.2.1 beschriebenen Aufbau. Der zentrale Unterschied besteht darin, dass mehrere U-Net-basierte Diffusionsmodelle mit unterschiedlicher Modellkapazität betrachtet werden. Die Modelle unterscheiden sich ausschließlich in kapazitätsbestimmenden Architekturparametern. Es gibt ein sehr kleines Modell (M1), ein kleines Modell (M2), ein mittleres Modell (M3) und ein großes Modell (M4). Die konkreten

Architekturen, einschließlich Kanaltiefen, Anzahl der ResNet- und Attention-Blöcke sowie der resultierenden Parameteranzahl sind in Tabelle 4.3 angegeben.

Eigenschaft	**M1**	**M2**	**M3**	**M4**
Gesamtparameter	129,721	714,860	3,868,452	13,673,284
Time-Embedding-Parameter	69,169	197,376	197,376	197,376
Channels Level 1	4	8	32	64
Channels Level 2	8	16	64	128
Channels Level 3	16	32	128	256
Channels Level 4	16	64	128	256
Attention L1	0	0	0	0
Attention L2	0	0	0	0
Attention L3	1	1	1	1
Attention L4	1	1	1	1
ResNet-Blöcke pro Tiefe	1	2	2	2

Tabelle 4.3.: Vergleich der architekturrelevanten Parameter der Diffusionsmodelle

Alle Modelle werden auf demselben parametrisierten Pseudoradar-Generator trainiert (vgl. Abschnitt 3.2), mit den festen Generatorparametern:

$$\lambda_{\text{Linien}} = 10, \quad \lambda_{\text{Punkte/Linie}} = 20, \quad \lambda_{\text{Clutter}} = 50.$$

Die erzeugten Punktwolken werden diskretisiert und dem jeweiligen Modell als Eingabe bereitgestellt (vgl. Abschnitt 3.1.1). Das Training aller Modellvarianten erfolgt mit identischen Hyperparametern. Die Modelle werden über denselben Zeitraum trainiert und zu festen Trainingszeitpunkten evaluiert. Zu diesen Zeitpunkten werden synthetische Samples generiert und die FRD_∞ wird zwischen den generierten Samples und unabhängigen Referenzdaten aus dem Pseudoradar-Generator berechnet.

Durch diesen Aufbau kann analysiert werden, wie sich die Modellkapazität auf die Konvergenzgeschwindigkeit der FRD_∞ sowie auf das erreichbare Qualitätsniveau der generierten Radar-Punktwolken auswirkt. Zusätzlich werden exemplarische Samples visualisiert, um qualitative Unterschiede zwischen den Modellvarianten zu veranschaulichen.

4.2.2.2. Ergebnisse

Abbildung 4.13 zeigt den Verlauf von FRD_∞ in Abhängigkeit vom Trainingsfortschritt für die vier betrachteten Diffusionsmodelle. In den ersten Trainingsschritten fällt das Distanzmaß bei allen Modellen deutlich ab. Dies weist darauf hin, dass bereits in frühen Iterationen grundlegende Eigenschaften der Zielverteilung erfasst werden. Dieses Verhalten lässt sich auch in Abbildung 4.14 beobachten, in der die eindimensionale Verteilung des Validitätsindikators der generierten Daten nach $5\,000$ Trainingsiterationen dargestellt ist. Die Zielverteilung des Pseudoradar-Generators in dieser Dimension ist durch eine stark dünnbesetzte Belegung des

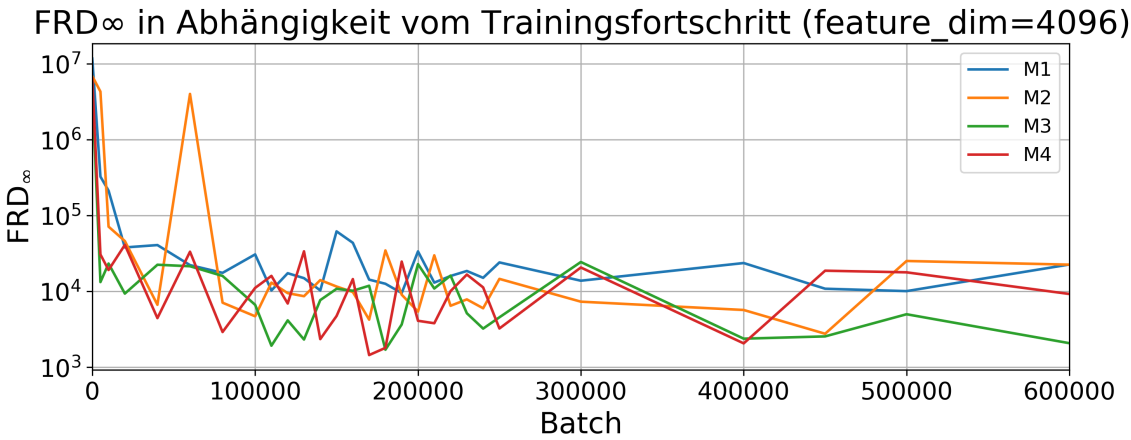

Abbildung 4.13.: Verlauf der FRD_∞ über den Trainingsfortschritt für Diffusionsmodelle mit unterschiedlicher Modellkapazität (vgl. Tabelle 4.3).

64×64-Gitters charakterisiert und weist im Erwartungswert 250 Einträge mit dem Wert 1 sowie $64 \cdot 64 - 250$ Einträge mit dem Wert 0 auf. Insbesondere die größeren Modelle M3 und M4 zeigen zu diesem frühen Trainingszeitpunkt bereits eine Verteilung des Validitätsindikators, die der Zielverteilung nahekommt. Dies äußert sich in zwei ausgeprägten Peaks bei 0 und 1. Die kleineren Modelle M1 und M2 weisen dagegen noch eine breite, in Richtung 1 verzogene Verteilung ohne klare Trennung der beiden Ausprägungen auf. Dieses unterschiedliche Verhalten spiegelt sich direkt im Verlauf der FRD_∞ wider, da die Modelle M3 und M4 in den frühen Trainingsiterationen eine deutlich steilere Abnahme zeigen.

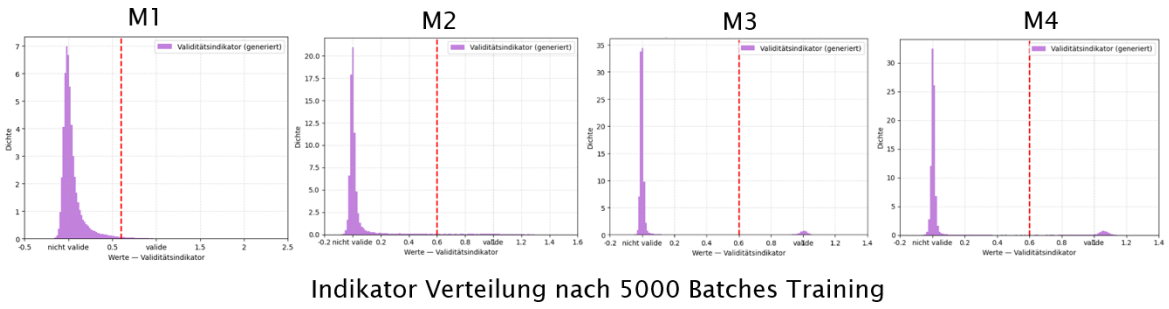

Abbildung 4.14.: Verteilung des Validitätsindikators der generierten Radar-Punktwolken nach $5\,000$ Trainingsiterationen für Diffusionsmodelle mit unterschiedlicher Modellkapazität (vgl. Tabelle 4.3). Die gestrichelte Linie markiert den Schwellenwert, ab dem eine Gitterzelle als aktiv (Punkt vorhanden) gilt.

Nach der initialen Trainingsphase weisen alle vier Modelle über den weiteren Trainingsverlauf eine erhöhte Varianz der FRD auf. Über den gesamten Trainingsverlauf hinweg erkennt man, dass das größte Modell (M4) den niedrigsten FRD-Wert erreicht. Die von den Modellen M2 und M3 erzielten FRD_∞-Werte liegen in einer ähnlichen Größenordnung wie die von M4. Das kleinste Modell (M1) unterschrei-

tet hingegen zu keinem Zeitpunkt einen FRD_∞-Wert von 10^4. Dieses Verhalten weist darauf hin, dass das Modell M1 die Zielverteilung nicht so gut wie die anderen Modelle approximiert. Diese erreichen alle mindestens einen FRD_∞-Wert von $5 \cdot 10^3$. Diese Beobachtung wird durch Abbildung 4.15 gestützt. Hier ist zu erkennen, dass die Modelle M2, M3 und M4 nach $600\,000$ Trainingsiterationen Punktwolken erzeugen, die sich entlang linearer Strukturen anordnen. Das Modell M1 erzeugt hingegen Punktwolken, bei denen keine linearen Strukturen erkennbar sind. Das weist darauf hin, dass die Modellkapazität von M1 nicht ausreicht, um die strukturellen Eigenschaften der durch die Pseudoradar-Generatoren definierten Zielverteilung konsistent zu reproduzieren.

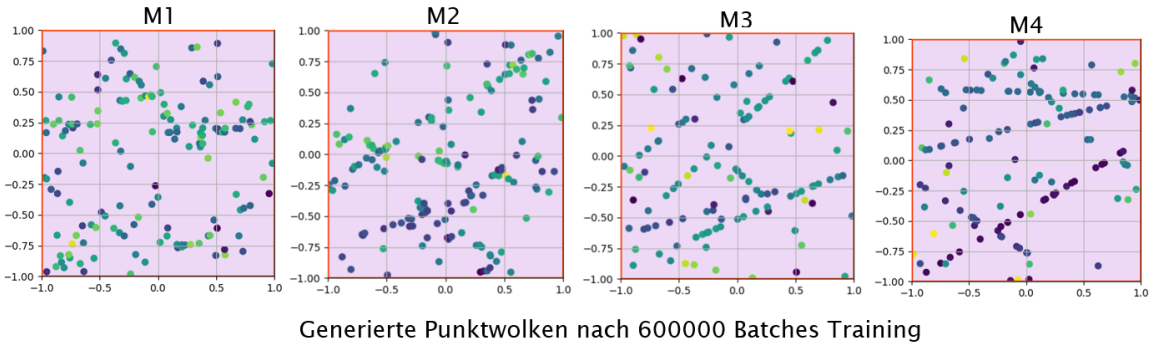

Abbildung 4.15.: Vergleich generierter Radar-Punktwolken nach $600\,000$ Trainingsiterationen für Diffusionsmodelle mit unterschiedlicher Modellkapazität (vgl. Tabelle 4.3).

Ergänzend zur Auswertung der FRD und der generierten Punktwolken zeigt Abbildung 4.16 den Verlauf der negativen Log-Likelihood für alle vier Modellvarianten. Auch hier ist bei allen Modellen ein starker initialer Abfall zu beobachten. Im Gegensatz zur FRD_∞ erreichen die Log-Likelihood-Werte jedoch bereits nach wenigen Trainingsschritten ein stabiles Niveau. Die Modelle M1, M3 und M4 konvergieren im Bereich von etwa 100000 Trainingsiterationen, während Modell M2 erst nach ungefähr 200000 Batches ein Plateau erreicht. Hinsichtlich der erreichten Grenzwerte konvergieren die mittleren und großen Modelle (M3 und M4) in der Log-Likelihood gegen den niedrigsten Wert. Das Modell M2 erreicht einen leicht erhöhten, jedoch vergleichbaren Grenzwert, während das kleinste Modell (M1) deutlich höhere Endwerte aufweist. Damit ist die Bewertung anhand der Log-Likelihood konsistent mit den Ergebnissen der FRD-Analyse sowie der qualitativen Betrachtung der generierten Samples.

Zusammenfassend zeigen die Ergebnisse, dass die Modellkapazität einen wesentlichen Einfluss auf das Trainingsverhalten und die erreichbare Sample-Qualität hat. Modelle mit zu geringer Kapazität (M1) sind nicht in der Lage, die strukturellen Eigenschaften der dünnbesetzten Radar-Punktwolken konsistent zu erfassen und konvergieren gegen höhere FRD_∞-Werte. Demgegenüber zeigen Modelle mit mittlerer Kapazität eine stabilere Konvergenz und erreichen ein hohes Qualitätsniveau, das mit dem der größten Modellvariante vergleichbar ist. Eine weitere Erhöhung

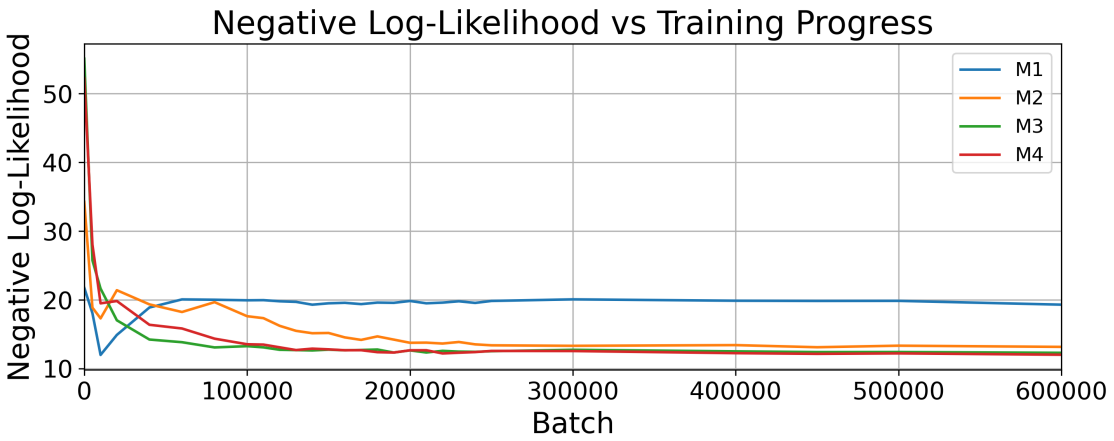

Abbildung 4.16.: Verlauf der negativen Log-Likelihood über den Trainingsfortschritt für Diffusionsmodelle mit unterschiedlicher Modellkapazität (vgl. Tabelle 4.3).

der Modellkapazität führt zwar zu einer etwas schnelleren Konvergenz, liefert jedoch keine Verbesserungen im erreichten Endniveau der FRD_{∞}.

Damit weisen die Ergebnisse auf einen Bereich ausreichender Modellkapazität hin, ab dem zusätzliche Parameter keinen signifikanten Qualitätsgewinn mehr erzeugen. Die Experimente beantworten somit **RQ4** und zeigen, dass die Modellkapazität sowohl die Konvergenzgeschwindigkeit als auch das erreichbare Qualitätsniveau der generierten Radar-Punktwolken maßgeblich beeinflusst.

5. Diskussion

Im folgenden Kapitel werden die Ergebnisse der durchgeführten Experimente diskutiert und im Hinblick auf die Forschungsfragen dieser Arbeit eingeordnet. Dabei werden die Ergebnisse in Bezug auf die Validierung der FRD und des Trainings der Diffusionsmodelle getrennt betrachtet. Anschließend werden die Limitationen dieser Arbeit aufgezeigt.

5.1. Diskussion der Ergebnisse zur Validierung der FRD (RQ1)

Die in Abschnitt 4.1.1 und Abschnitt 4.1.2 beschriebenen Experimente untersuchen, ob die FRD in kontrollierten Szenarien die erwarteten Eigenschaften eines Distanzmaßes erfüllt (**RQ1**). Aus den Ergebnissen lassen sich zwei zentrale Aussagen ableiten. Erstens zeigt die FRD bei identischen Verteilungen das erwartete Grenzverhalten, und zweitens reagiert sie konsistent auf Veränderungen der zugrundeliegenden Datenverteilungen.

Die Konvergenz der FRD gegen Null bei identischen Verteilungen zeigt eine notwendige Konsistenzeigenschaft, die für den späteren Einsatz als Validierungsmetrik relevant ist. Das Ergebnis, dass Null innerhalb der Konfidenzintervalle der extrapolierten FRD_∞-Schätzungen liegt, spricht dafür, dass verbleibende Abweichungen bei endlichen Stichproben primär durch Schätzfehler und numerische Approximationen verursacht werden. Damit wird zugleich die Rolle der Extrapolation motiviert, da reine Punkt-Schätzungen der FRD bei endlichem n von ihrem Grenzwert abweichen können, was sie in den Experimenten auch tun. Zusätzlich zu den experimentellen Ergebnissen bestätigt auch die theoretische Analyse der FRD, dass sie für identische Verteilungen gegen Null konvergiert.

Über das reine Grenzverhalten hinaus ist auch der Einfluss der Projektionsdimension auf die Distanzwerte der FRD relevant. Das beobachtete Skalierungsgesetz ist plausibel, da die Anzahl der Freiheitsgrade der Kovarianzschätzung quadratisch mit der Dimension wächst und dadurch die Varianz der geschätzten Kovarianzstruktur bei endlichen Stichproben ansteigt. Für praktische Anwendungen folgt daraus, dass FRD-Werte nur bei konsistent gewählter Projektionsdimension direkt vergleichbar sind oder alternativ eine Normalisierung erforderlich ist.

Die Experimente aus Abschnitt 4.1.2 haben gezeigt, dass die FRD_∞ für unterschiedliche Verteilungen positive Werte annimmt und damit Unterschiede zwischen Generatoren abbildet. Die beobachtete Korrelation mit der approximierten Log-Likelihood deutet darauf hin, dass die FRD die Ähnlichkeit zwischen Radar-Daten konsistent abbildet. Die Abweichungen, die trotzdem zwischen den Bewertungen der Log-Likelihood und der FRD bestehen, sind darauf zurückzuführen, dass beide Distanzmaße unterschiedliche Eigenschaften bewerten. Die Log-Likelihood bewertet die Daten im Sinne des zugrundeliegenden Modells, während die FRD ausschließlich auf Momentenunterschieden im projizierten Feature-Raum beruht.

Insgesamt stützen die Ergebnisse, dass die FRD in kontrollierten Pseudoradar-Szenarien eine geeignete Validierungsmetrik unter den betrachteten Bedingungen darstellt (**RQ1**).

5.2. Generierung von Radar-Daten mit Diffusionsmodellen (RQ2-RQ4)

In den Experimenten in Abschnitt 4.2.1 und Abschnitt 4.2.2 wurde der Einsatz von U-Net-basierten Diffusionsmodellen zur Generierung von Radar-Daten untersucht. Der Fokus lag darauf zu prüfen, ob diese Modelle in der Lage sind, die dünnbesetzte Struktur von Radar-Punktwolken zu generieren (**RQ2**). Zusätzlich wurde analysiert, ob die FRD den Trainingsverlauf konsistent in Bezug auf die Sample-Qualität und die Log-Likelihood als Referenzmetrik, widerspiegelt (**RQ3**). Abschließend wurden Diffusionsmodelle mit unterschiedlicher Modellkapazität auf die Generierung von Radar-Daten trainiert und hinsichtlich der erreichten Sample-Qualität sowie der resultierenden FRD-Werte verglichen (**RQ4**).

Die Analyse legt nahe, dass U-Net-basierte Diffusionsmodelle in der Lage sind, die charakteristische Struktur dünnbesetzter Radar-Punktwolken zu generieren. Die generierten Daten weisen eine vergleichbare Dünnbesetztheit auf und reproduzieren die dominanten linearen Strukturen der Referenzdaten. Diese Annäherung an die Zielverteilung wird durch die im Trainingsverlauf abnehmende und schließlich stabil konvergierende FRD_∞ gestützt. Damit wird für die betrachteten synthetischen Szenarien gezeigt, dass Diffusionsmodelle grundsätzlich geeignet sind, dünnbesetzte und verrauschte Daten zu modellieren.

Das Kapazitätsexperiment in Abschnitt 4.2.2 zeigt, dass die Modellkapazität einen wesentlichen Einfluss auf die Generierung dünnbesetzter Radar-Punktwolken hat. Modelle mit zu geringer Kapazität sind nicht in der Lage, die strukturellen Eigenschaften der Zielverteilung konsistent zu erfassen und konvergieren gegen höhere FRD_∞-Werte. Demgegenüber erreichen Modelle mit mittlerer und hoher Kapazität eine vergleichbare Sample-Qualität sowie ähnliche FRD-Werte. Eine weitere Erhöhung der Modellkapazität über dieses Niveau hinaus führt nicht zu

einem signifikanten Qualitätsgewinn, sondern wirkt sich lediglich auf die Konvergenzgeschwindigkeit aus. Das größte Modell konvergiert zwar schneller, das langfristig erreichte FRD_∞-Niveau unterscheidet sich jedoch nur geringfügig von dem der mittleren Modellvariante. Dabei ist zu berücksichtigen, dass alle Modelle während des Trainings auf einem Datenstrom aus dem Pseudoradar-Generator trainiert wurden. Dadurch, dass keine endliche Trainingsmenge verwendet wurde, kann kein Overfitting im Sinne einer Überanpassung an die Trainingsdaten auftreten. Die beobachteten Unterschiede zwischen den Modellvarianten sind daher auf ihre unterschiedliche Repräsentationsfähigkeit und nicht auf Überanpassungseffekte zurückzuführen. In einem Szenario mit endlichen Trainingsdaten kann eine Erhöhung der Modellkapazität zu Overfitting führen. Die Ergebnisse sind daher als eine asymptotische Aussage über die effektive Modellkapazität der betrachteten Diffusionsmodelle zu interpretieren. Sie beschreiben, welche strukturelle Komplexität der Zielverteilung von Modellen mit einer gegebenen Anzahl an Parametern prinzipiell repräsentiert werden kann, unabhängig von Effekten endlicher Trainingsdaten.

Zusammenfassend zeigen die Experimente, dass das verwendete Diffusionsmodell die Zielverteilung der synthetischen Pseudoradar-Daten approximieren kann (**RQ2**). Gleichzeitig legt der beobachtete Trainingsverlauf nahe, dass die FRD geeignet ist, die qualitative Entwicklung der generierten Samples während des Trainings zu bewerten (**RQ3**). Darüber hinaus zeigen die Ergebnisse, dass für die betrachteten synthetischen Radar-Daten ein Bereich ausreichender Modellkapazität existiert. Ab dieser Modellkapazität führen zusätzliche Parameter zu keiner Erhöhung der Qualität der generierten Daten (**RQ4**).

5.3. Limitationen

5.3.1. Verwendung ausschließlich synthetischer Radar-Daten

Eine zentrale Limitation dieser Arbeit besteht darin, dass alle Experimente ausschließlich auf synthetisch erzeugten Radar-Daten basieren. Reale Radar-Messdaten weisen zusätzliche Effekte wie Sensorrauschen, Mehrwegeffekte, Okklusionen und nicht-stationäre Hintergrundstrukturen auf, die in den verwendeten Pseudoradar-Szenarien nur vereinfacht modelliert werden. Die empirischen Ergebnisse sind daher primär für kontrollierte, idealisierte Bedingungen gültig.

Die Verwendung synthetischer Daten ist dabei nicht allein als Schwäche zu sehen, sondern stellt eine bewusst gewählte methodische Entscheidung dar. Ziel der Arbeit war es, die grundlegenden Eigenschaften der FRD isoliert zu untersuchen und ihre Konsistenz sowie Sensitivität unter exakt kontrollierbaren Verteilungsänderungen zu analysieren. Synthetische Pseudoradar-Generatoren ermöglichen es, einzelne Parameter gezielt zu variieren und deren Einfluss auf das Distanzmaß

unabhängig von unbeobachtbaren Störeinflüssen zu untersuchen. Eine vergleichbare Analyse wäre mit realen Radar-Daten nur eingeschränkt möglich, da dort die Verteilungen und Einflussfaktoren nicht explizit kontrollierbar sind. Somit stellt die Beschränkung auf synthetische Radar-Daten eine Abstraktion dar, die die Aussagekraft für echte Radar-Daten einschränkt, gleichzeitig aber auch die Grundlage für die Evaluation der FRD ist.

In Bezug auf das Training der Diffusionsmodelle ist die Limitation bei der Verwendung von Pseudoradar-Generatoren größer. Zwar wurde gezeigt, dass die eingesetzten Diffusionsmodelle in der Lage sind, die grundlegende dünnbesetzte Struktur der verwendeten Pseudoradar-Generatoren zu reproduzieren, jedoch lässt sich aus diesen Ergebnissen keine Aussage über die Übertragbarkeit auf reale Radar-Daten ableiten. Reale Radar-Messdaten entstehen aus komplexen physikalischen Wechselwirkungen und weisen neben linearen auch nicht-lineare Strukturen auf, die in den synthetischen Szenarien nicht abgebildet werden. Darüber hinaus besteht eine Domänenlücke zwischen synthetisch erzeugten und realen Radar-Daten. Die Pseudoradar-Generatoren definieren glatte, parametrisierte Zielverteilungen, während reale Radar-Daten durch Sensorrauschen, Mehrwegeffekte, materialspezifische Reflexionen sowie zeitlich und räumlich nicht-stationäre Prozesse geprägt sind (siehe Abschnitt 2.1). Es kann daher nicht ausgeschlossen werden, dass die Diffusionsmodelle primär generatorspezifische Artefakte erlernen, die in realen Radar-Szenarien nicht auftreten. Diese Einschränkungen sind jedoch im Kontext des Ziels einzuordnen, das mit dem Training der Diffusionsmodelle verfolgt wurde. Ziel der Experimente war es nicht, ein generatives Modell zu entwickeln, das reale Radar-Daten realistisch abbildet, sondern zu zeigen, dass Diffusionsmodelle grundsätzlich in der Lage sind, dünnbesetzte Radar-Daten zu generieren. Die Ergebnisse beziehen sich somit ausschließlich auf synthetische Radar-Daten und erlauben keine direkte belegbare Aussage über die Leistungsfähigkeit der Modelle in realen Radar-Szenarien. Sie liefern jedoch Hinweise darauf, dass der Ansatz prinzipiell funktioniert.

5.3.2. Verwenden von Datenströmen

Eine weitere Limitation der Arbeit ergibt sich aus der Verwendung kontinuierlicher Datenströme für das Training der Diffusionsmodelle. Alle Diffusionsmodelle wurden auf einem unbegrenzten Datenstrom aus dem Pseudoradar-Generator trainiert. Dadurch stehen für jeden Trainingsschritt neue Stichproben aus der Zielverteilung zur Verfügung. Somit ist es möglich eine asymptotische Analyse der effektiven Modellkapazität durchzuführen und eine Überanpassung an einzelne Trainingssamples auszuschließen. Für reale Anwendungen ist dieses Szenario nicht repräsentativ. In echten Anwendungen steht lediglich eine endliche, oft begrenzte Menge an Trainingsdaten zur Verfügung. In diesen Szenarien kann die Modellkapazität nicht unabhängig von der Datenmenge betrachtet werden, da eine zu hohe Modellkapazität zu Overfitting und eingeschränkter Generalisierungsfähigkeit führen kann. Die in dieser Arbeit gewonnenen Aussagen beziehen

sich daher explizit auf idealisierte Trainingsbedingungen mit unbegrenzter Datenverfügbarkeit. Sie erlauben keine direkte Aussage darüber, welche Modellgröße unter realistischen Datenszenarien optimal ist, sondern beschreiben vielmehr die asymptotische Repräsentationsfähigkeit der betrachteten Diffusionsmodelle. Eine Übertragung der Ergebnisse auf reale Anwendungen erfordert daher zusätzliche Experimente mit endlichen Datensätzen.

5.3.3. Fehlende task-basierte Validierung

Eine task-basierte Evaluation wäre notwendig, um die Aussagekraft der FRD im Hinblick auf reale Anwendungsziele zu beurteilen. Generative Modelle werden schlussendlich dafür eingesetzt, die Trainingsdaten von Klassifikatoren zu vervollständigen (vgl. Abschnitt 1.1). Die Klassifikatoren sollen dadurch verbessert werden. Daraus ergibt sich eine weitere Limitation dieser Arbeit: Es wurde nicht untersucht, inwieweit die FRD mit dem praktischen Nutzen der generierten Daten für nachgelagerte Aufgaben korreliert. Damit bleibt offen, ob eine geringe Distanz im Sinne der FRD zwangsläufig mit einer hohen Aufgabenrelevanz der generierten Daten einhergeht. Es ist möglich, dass generierte Samples zwar statistisch nah an der Referenzverteilung liegen, jedoch für konkrete Aufgaben nur eingeschränkt informative oder sogar irreführende Strukturen enthalten. Die Korrelation der FRD mit dem praktischen Nutzen hätte zusätzliche Erkenntnisse über den praktischen Mehrwert der Metrik geliefert, lag jedoch außerhalb des Umfangs dieser Arbeit.

6. Fazit und Ausblick

In diesem Kapitel werden die zentralen Ergebnisse der Arbeit zusammengefasst und im Rahmen eines Ausblicks mögliche weiterführende Forschungsrichtungen dargestellt.

6.1. Fazit

Diese Arbeit verfolgte zwei zentrale Ziele. Das erste Ziel bestand in der Vorstellung und Evaluation einer domänenspezifischen Metrik zur quantitativen Bewertung synthetischer Radar-Daten. Motiviert durch die eingeschränkte Übertragbarkeit etablierter Bildmetriken auf Radar-Punktwolken wurde mit der FRD ein Distanzmaß eingeführt, das auf der FD basiert und zufällige Projektionen als Feature-Extraktor verwendet.

Die theoretische Analyse sowie die durchgeführten Experimente in kontrollierten Szenarien zeigen, dass die FRD zentrale Eigenschaften eines geeigneten Distanzmaßes erfüllt. Insbesondere konvergiert die FRD für identische Verteilungen mit wachsender Stichprobengröße gegen Null und reagiert konsistent auf Veränderungen der Datenverteilungen. Darüber hinaus konnte gezeigt werden, dass die Projektionsdimension einen Einfluss auf die absoluten Distanzwerte hat, was bei der praktischen Anwendung der Metrik berücksichtigt werden muss. Im Vergleich mit der approximierten Log-Likelihood liefert die FRD konsistente Rangfolgen der betrachteten Generatoren. Damit wurde gezeigt, dass die FRD zur Validierung synthetischer Radar-Daten in kontrollierten Pseudoradar-Szenarien geeignet ist. Die vorliegenden Ergebnisse schaffen eine methodische Grundlage für die quantitative Bewertung generativer Modelle im Kontext von Radar-Daten. Auf dieser Basis lassen sich in zukünftigen Arbeiten weiterführende Fragestellungen untersuchen, die über die in dieser Arbeit betrachteten, synthetischen Szenarien hinausgehen.

Das zweite Ziel dieser Arbeit bestand darin zu untersuchen, ob Diffusionsmodelle auf U-Net-Basis grundsätzlich in der Lage sind, dünnbesetzte Radar-Punktwolken zu generieren. Die experimentellen Ergebnisse zeigen, dass die betrachteten Diffusionsmodelle die charakteristische dünnbesetzte Struktur der synthetischen Pseudoradar-Daten reproduzieren können und im Verlauf des Trainings die durch den Generator vorgegebenen Strukturen erzeugen. Diese qualitative Annäherung an die Zielverteilung spiegelt sich auch in der Konvergenz der FRD wider. Ergänzend konnte gezeigt werden, dass die Modellkapazität einen wesentlichen

Einfluss auf die erreichbare Sample-Qualität hat. Während Modelle mit zu geringer Kapazität die strukturellen Eigenschaften der Zielverteilung nicht zuverlässig erfassen, erreichen Modelle mit ausreichender Kapazität stabile und vergleichbare Qualitätsniveaus. Damit liefern die Ergebnisse einen empirischen Nachweis dafür, dass Diffusionsmodelle grundsätzlich unter den betrachteten synthetischen Bedingungen für die Generierung dünnbesetzter Radar-Daten geeignet sind.

Insgesamt stellt diese Arbeit einen weiteren Schritt hin zu einer allgemein akzeptierten, domänenspezifischen Distanzmetrik für die Bewertung synthetischer Radar-Daten dar. Darüber hinaus deuten die Ergebnisse darauf hin, dass U-Net-Diffusionsmodelle in der Lage sind, Radar-Daten zu erzeugen, und damit eine Grundlage für den zukünftigen Einsatz generativer Modelle als integralen Bestandteil radarbasierter Datenpipelines schaffen kann.

6.2. Ausblick

Die in dieser Arbeit vorgestellte und evaluierte FRD zeigt in kontrollierten Experimenten konsistentes und erwartungskonformes Verhalten. Aufbauend auf den Ergebnissen ergeben sich mehrere Ansatzpunkte für zukünftige Arbeiten, die sowohl eine praktische Erweiterung als auch eine Verallgemeinerung der vorgestellten Methodik betreffen.

6.2.1. Anwendung der FRD auf reale Radar-Daten

Ein nächster Schritt ist die Analyse der Eigenschaften der FRD bei realen Radar-Daten. Insbesondere der Einfluss von Messrauschen, unvollständigen Punktwolken sowie unterschiedlicher Sensorcharakteristiken auf die Stabilität der Metrik stellen eine relevante Fragestellung dar. Eine solche Untersuchung könnte Aufschluss darüber geben, inwieweit die FRD robust gegenüber realweltlichen Störeinflüssen ist.

Aufbauend auf der Analyse stellt das Training von Diffusionsmodellen auf realen Radar-Daten einen naheliegenden nächsten Schritt dar. Die generierten Radar-Daten können mithilfe der FRD mit realen Referenzdaten verglichen und dadurch ihre Qualität validiert werden. Zusätzlich bietet sich speziell im Automobil-Kontext die konditionierte Erzeugung von Radar-Daten auf Basis von Bilddaten an. In modernen Fahrzeugsystemen werden Radar- und Kamerasensoren typischerweise gemeinsam eingesetzt, sodass isolierte generative Modelle für einzelne Sensordaten nur eingeschränkten praktischen Nutzen besitzen. Diffusionsmodelle auf U-Net-Basis haben sich im Bereich der Bildgenerierung bereits als geeignet für konditionierte Generierungsaufgaben erwiesen, etwa durch die Konditionierung auf Text oder andere Bilder [130, 131]. Da in dieser Arbeit gezeigt wurde, dass Diffusionsmodelle in der Lage sind, strukturierte Radar-Daten zu erzeugen, liegt

die Annahme nahe, dass sich dieser Ansatz auch auf die bildkonditionierte Generierung von Radar-Daten übertragen lässt. Die FRD kann in diesem Szenario weiterhin als quantitatives Evaluationsmaß dienen, um die Qualität und Konsistenz der konditioniert erzeugten Radar-Daten zu bewerten.

6.2.2. Übertragung des Ansatzes auf Bilddaten mittels Wavelet-Darstellung

Eine Erweiterung ergibt sich aus der Beobachtung, dass die in dieser Arbeit vorgestellte FRD konzeptionell nicht auf Radar-Daten beschränkt ist. Zwar ist das eingeführte Diskretisierungsverfahren speziell auf die Verarbeitung von Punktwolken ausgelegt und für andere Datenformate ungeeignet, jedoch existieren für diese vergleichbare Transformationen. Ein Beispiel hierfür ist die Wavelet-Transformation für Bilddaten [132]. Diese erlaubt eine kompakte und dünnbesetzte Darstellung von Bildinformationen, indem sie strukturelle Inhalte wie Kanten, Texturen und lokale Frequenzanteile explizit erfasst. Die resultierende Mehrskalenrepräsentation weist ähnliche Eigenschaften wie die verwendete Gitterstruktur. Dadurch wird die Voraussetzung für den Einsatz zufälliger Projektionen als Feature-Extraktor geschaffen. In Kombination mit zufälligen Projektionen könnte auf diese Weise ein trainingsunabhängiger Merkmalsraum für Bilddaten definiert werden. Analog zum Ansatz von Veeramacheneni et al. [132] entsteht dadurch eine domänenunabhängige Distanzmetrik. Sie würde ohne vortrainierte Klassifikationsmodelle auskommen. Das spielt vor allem in Szenarien, in denen die betrachteten Bilder stark vom ImageNet-Datensatz abweichen eine wichtige Rolle. Hier könnten die Features, die das Inception-Netz liefert, als Bewertungsmaß ungeeignet sein.

Insgesamt eröffnet die in dieser Arbeit vorgestellte Methodik die Perspektive, die FRD als allgemeines, domänenübergreifendes Framework zur Bewertung generativer Modelle zu etablieren.

Abkürzungsverzeichnis

CNN Convolutional Neural Network

DDIM Denoising Diffusion Implicit Model

DDPM Denoising Diffusion Probabilistic Model

ELBO Evidence Lower Bound

EMA Exponential Moving Average

FD Fréchet-Distance

FID Fréchet-Inception-Distance

FRD Fréchet-Radar-Distance

GAN Generative Adversarial Network

KLD Kullback-Leibler-Divergenz

LiDAR Light Detection and Ranging

LSAP Linear-Sum-Assignment-Problem

ML Maschinelles Lernen

MSE Mean Squared Error

PPP Public-Private-Partnership

RAD-Tensor Range-Azimut-Doppler-Tensor

Radar Radio Detection and Ranging

RIC Restricted-Isometry-Konstante

RIP Restricted-Isometry-Property

VAE Variational Autoencoder

Abbildungsverzeichnis

Hinweis: Sämtliche Abbildungen in dieser Arbeit wurden eigenständig erstellt. Die in den Abbildungen genannten Quellen geben lediglich an, dass die Abbildung aus der Quelle reproduziert oder als Inspiration genutzt wurde.

Literaturverzeichnis

[1] M. Maurer, „Einleitung,“ in *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte*, M. Maurer, J. C. Gerdes, B. Lenz und H. Winner, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, S. 1–8. DOI: 10.1007/978-3-662-45854-9_1.

[2] J. Van Brummelen, M. O’Brien, D. Gruyer und H. Najjaran, „Autonomous vehicle perception: The technology of today and tomorrow,“ *Transportation Research Part C: Emerging Technologies*, Jg. 89, S. 384–406, 2018, ISSN: 0968-090X. DOI: 10.1016/j.trc.2018.02.012.

[3] World Health Organization, *Global Status Report on Road Safety 2023*, Zugriff am: 2025-09-16, 2023. Adresse: https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023.

[4] T. Winkle, „Sicherheitspotenzial automatisierter Fahrzeuge: Erkenntnisse aus der Unfallforschung,“ in *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte*, M. Maurer, J. C. Gerdes, B. Lenz und H. Winner, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, S. 351–376. DOI: 10.1007/978-3-662-45854-9_17.

[5] Q. Song, K. Tan, P. Runeson und S. Persson, „Critical scenario identification for realistic testing of autonomous driving systems,“ *Software Quality Journal*, Jg. 31, Nr. 2, S. 441–469, 2023. DOI: 10.1007/s11219-022-09604-2.

[6] E. Yurtsever, J. Lambert, A. Carballo und K. Takeda, „A Survey of Autonomous Driving: Common Practices and Emerging Technologies,“ *IEEE Access*, Jg. 8, S. 58 443–58 469, 2020. DOI: 10.1109/ACCESS.2020.2983149.

[7] M. Schwonberg u. a., „Survey on Unsupervised Domain Adaptation for Semantic Segmentation for Visual Perception in Automated Driving,“ *IEEE Access*, Jg. 11, S. 54 296–54 336, 2023. DOI: 10.1109/ACCESS.2023.3277785.

[8] W. Ertel, *Introduction to Artificial Intelligence*, 3rd. Cham: Springer, 2021. DOI: 10.1007/978-3-658-26763-6.

[9] I. Goodfellow, Y. Bengio und A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[10] S. Shafaei, S. Kugele, M. H. Osman und A. Knoll, „Uncertainty in Machine Learning: A Safety Perspective on Autonomous Driving," in *Computer Safety, Reliability, and Security*, B. Gallina, A. Skavhaug, E. Schoitsch und F. Bitsch, Hrsg., Cham: Springer International Publishing, 2018, S. 458–464, ISBN: 978-3-319-99229-7.

[11] P. Liu, Y. Chen, F. Yu und Q. Zhang, „Mastering adverse weather: a two-stage approach for robust semantic segmentation in autonomous driving," *The Visual Computer*, Jg. 41, Nr. 6, S. 4321–4346, 2025. DOI: 10.1007/s00371-024-03663-1.

[12] C. Qian u. a., „AllWeather-Net: Unified Image Enhancement for Autonomous Driving Under Adverse Weather and Low-Light Conditions," in *Pattern Recognition*, A. Antonacopoulos, S. Chaudhuri, R. Chellappa, C.-L. Liu, S. Bhattacharya und U. Pal, Hrsg., Cham: Springer Nature Switzerland, 2025, S. 151–166, ISBN: 978-3-031-78113-1.

[13] OpenAI, *DALL·E 3*, Zugriff am: Sep. 26, 2025, 2023. Adresse: https://openai.com/index/dall-e-3/.

[14] T. Lesort, A. Stoian, J.-F. Goudou und D. Filliat, „Training Discriminative Models to Evaluate Generative Ones," in *Artificial Neural Networks and Machine Learning – ICANN 2019: Image Processing*, I. V. Tetko, V. Kůrková, P. Karpov und F. Theis, Hrsg., Cham: Springer International Publishing, 2019, S. 604–619, ISBN: 978-3-030-30508-6.

[15] M. N. Vivekananda, P. Ashok Shidlyali und V. V. Malgi, „Generative AI Meets Large Language Models: Evolution, Challenges, and Future Directions," in *2025 Global Conference in Emerging Technology (GINOTECH)*, 2025, S. 1–6. DOI: 10.1109/GINOTECH63460.2025.11076892.

[16] Z. You, X. Zhang, H. Guo, J. Wang und C. Li, „Are Images Indistinguishable to Humans Also Indistinguishable to Classifiers?" In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, S. 28 790–28 800. DOI: 10.1109/CVPR52734.2025.02681.

[17] L. Wu, B. Trippe, C. Naesseth, D. Blei und J. P. Cunningham, „Practical and Asymptotically Exact Conditional Sampling in Diffusion Models," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt und S. Levine, Hrsg., Bd. 36, Curran Associates, Inc., 2023, S. 31 372–31 403. Adresse: https://proceedings.neurips.cc/paper_files/paper/2023/file/63e8bc7bbf1cfea36d1d1b6538aecce5-Paper-Conference.pdf.

[18] L. Xu, M. Skoularidou, A. Cuesta-Infante und K. Veeramachaneni, „Modeling Tabular data using Conditional GAN," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox und R. Garnett, Hrsg., Bd. 32, Curran Associates, Inc., 2019. Adresse:

https://proceedings.neurips.cc/paper_files/paper/2019/file/254ed7d2de3b23ab10936522dd547b78-Paper.pdf.

[19] E. Betzalel, C. Penso und E. Fetaya, „Evaluation Metrics for Generative Models: An Empirical Study," *Machine Learning and Knowledge Extraction*, Jg. 6, Nr. 3, S. 1531–1544, 2024. DOI: 10.3390/make6030073.

[20] M. Arjovsky, S. Chintala und L. Bottou, „Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, Ser. ICML'17, Sydney, NSW, Australia: JMLR.org, 2017, S. 214–223.

[21] M. Stenger, R. Leppich, I. Foster, S. Kounev und A. Bauer, „Evaluation is key: a survey on evaluation measures for synthetic time series," *Journal of Big Data*, Jg. 11, Nr. 1, S. 66, 2024. DOI: 10.1186/s40537-024-00924-7.

[22] S. Jayasumana, S. Ramalingam, A. Veit, D. Glasner, A. Chakrabarti und S. Kumar, „Rethinking FID: Towards a Better Evaluation Metric for Image Generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Juni 2024, S. 9307–9315.

[23] S. Zhou, M. Gordon, R. Krishna, A. Narcomey, L. F. Fei-Fei und M. Bernstein, „HYPE: A Benchmark for Human eYe Perceptual Evaluation of Generative Models," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox und R. Garnett, Hrsg., Bd. 32, Curran Associates, Inc., 2019. Adresse: https://proceedings.neurips.cc/paper_files/paper/2019/file/65699726a3c601b9f31bf04019c8593c-Paper.pdf.

[24] F. Cheema und R. Urner, „Precision Recall Cover: A Method For Assessing Generative Models," in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, F. Ruiz, J. Dy und J.-W. van de Meent, Hrsg., Ser. Proceedings of Machine Learning Research, Bd. 206, PMLR, Apr. 2023, S. 6571–6594. Adresse: https://proceedings.mlr.press/v206/cheema23a.html.

[25] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler und S. Hochreiter, „GANs trained by a two timescale update rule converge to a local nash equilibrium," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Ser. NIPS'17, Long Beach, California, USA: Curran Associates Inc., 2017, S. 6629–6640, ISBN: 9781510860964.

[26] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford und X. Chen, „Improved techniques for training GANs," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Ser. NIPS'16, Barcelona, Spain: Curran Associates Inc., 2016, S. 2234–2242.

[27] Y. Benny, T. Galanti, S. Benaim und L. Wolf, „Evaluation Metrics for Conditional Image Generation,“ *International Journal of Computer Vision*, Jg. 129, Nr. 5, S. 1712–1731, 2021, ISSN: 1573-1405. DOI: 10.1007/s11263-020-01424-w.

[28] M. J. Chong und D. Forsyth, „Effectively Unbiased FID and Inception Score and Where to Find Them,“ in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, S. 6069–6078. DOI: 10.1109/CVPR42600.2020.00611.

[29] I. Goodfellow u. a., „Generative Adversarial Nets,“ in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[30] D. P. Kingma und M. Welling, „Auto-Encoding Variational Bayes,“ in *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014. Adresse: https://openreview.net/forum?id=33X9fd2-9FyZd.

[31] J. Ho, A. Jain und P. Abbeel, „Denoising diffusion probabilistic models,“ in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Ser. NIPS ’20, Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN: 9781713829546.

[32] J. Song, C. Meng und S. Ermon, „Denoising Diffusion Implicit Models,“ in *International Conference on Learning Representations*, 2021. Adresse: https://openreview.net/forum?id=St1giarCHLP.

[33] R. Zhang, D. Xue, Y. Wang, R. Geng und F. Gao, „Towards Dense and Accurate Radar Perception via Efficient Cross-Modal Diffusion Model,“ *IEEE Robotics and Automation Letters*, Jg. 9, Nr. 9, S. 7429–7436, 2024. DOI: 10.1109/LRA.2024.3426389.

[34] A. Tuel, T. Kerdreux, C. Hulbert und B. Rouet-Leduc, *Diffusion Models for Interferometric Satellite Aperture Radar*, 2023. arXiv: 2308.16847 [cs.CV]. Adresse: https://arxiv.org/abs/2308.16847.

[35] S. Wang, X. Luo, Y. Xie und W. Wang, „SDDiff: boosting radar perception via spatial-doppler diffusion,“ in *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, Ser. IJCAI ’25, Montreal, Canada, 2025, ISBN: 978-1-956792-06-5. DOI: 10.24963/ijcai.2025/979.

[36] K. Luan, C. Shi, N. Wang, Y. Cheng, H. Lu und X. Chen, „Diffusion-Based Point Cloud Super-Resolution for mmWave Radar Data,“ in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, S. 11 171–11 177. DOI: 10.1109/ICRA57147.2024.10611026.

[37] J. Neuhöfer und J. Reichardt, „Towards Measuring the Realism of Generative Models for Radar Data,“ Internes Poster, nicht veröffentlicht, 2025.

[38] nxtAIM Konsortium, *NXT GEN AI METHODS - Generative Methoden für Perzeption, Prädiktion und Planung*, `https://nxtaim.de/`, Zugriff am 18.09.2025, 2024.

[39] nxtAIM Konsortium, *NXT GEN AI METHODS - Generative Methoden für Perzeption, Prädiktion und Planung - Projektbeschreibung*, `https://nxtaim.de/projekt/`, Zugriff am 18.09.2025, 2024.

[40] J. Gamba, „Fundamentals of Radar Systems,“ in *Radar Signal Processing for Autonomous Driving*. Singapore: Springer Nature Singapore, 2020, S. 1–14. DOI: 10.1007/978-981-13-9193-4_1.

[41] B. I. Bakare, M. U. Ajaegbu und V. E. Idigo, „A Comprehensive Review of Radar System Technology,“ *Quest Journals: Journal of Electronics and Communication Engineering Research*, Jg. 8, Nr. 8, S. 16–23, 2022, ISSN: 2321-5941.

[42] A. Srivastav und S. Mandal, „Radars for Autonomous Driving: A Review of Deep Learning Methods and Challenges,“ *IEEE Access*, Jg. 11, S. 97 147–97 168, 2023. DOI: 10.1109/ACCESS.2023.3312382.

[43] F. Engels, P. Heidenreich, M. Wintermantel, L. Stäcker, M. Al Kadi und A. M. Zoubir, „Automotive Radar Signal Processing: Research Directions and Practical Challenges,“ *IEEE Journal of Selected Topics in Signal Processing*, Jg. 15, Nr. 4, S. 865–878, 2021. DOI: 10.1109/JSTSP.2021.3063666.

[44] C. Waldschmidt, J. Hasch und W. Menzel, „Automotive Radar — From First Efforts to Future Systems,“ *IEEE Journal of Microwaves*, Jg. 1, Nr. 1, S. 135–148, 2021. DOI: 10.1109/JMW.2020.3033616.

[45] C. Waldschmidt, R. Weigel, T. Jaeschke und T. Zwick, *Millimeter Wave Radar: Hardware and Signal Processing*. Springer International Publishing, 2024, Open Access. DOI: 10.1007/978-3-031-89118-2.

[46] D. Hunt u. a., „RadCloud: Real-Time High-Resolution Point Cloud Generation Using Low-Cost Radars for Aerial and Ground Vehicles,“ in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, S. 12 269–12 275. DOI: 10.1109/ICRA57147.2024.10610839.

[47] S. M. Patole, M. Torlak, D. Wang und M. Ali, „Automotive radars: A review of signal processing techniques,“ *IEEE Signal Processing Magazine*, Jg. 34, Nr. 2, S. 22–35, 2017. DOI: 10.1109/MSP.2016.2628914.

[48] H. Gim u. a., „Suitability of Various Lidar and Radar Sensors for Application in Robotics: A Measurable Capability Comparison,“ *IEEE Robotics and Automation Magazine*, Jg. 30, Nr. 3, S. 28–43, 2023. DOI: 10.1109/MRA.2022.3188213.

[49] S. Agrawal und G. B. Khairnar, „A Comparative Assessment of Remote Sensing Imaging Techniques: Optical, SAR and LiDAR,“ *Personal and Ubiquitous Computing*, 2023, Open Access. DOI: 10.1007/s00779-023-01736-x.

[50] T. Fei, „Automotive Radar Systems: Architecture, Signal Processing, and Future Perspectives,“ in *Vehicle Technology and Automotive Engineering*, H. Koten und S. Sarikoç, Hrsg., Rijeka: IntechOpen, 2025, Kap. 2. DOI: 10.5772/intechopen.1008976.

[51] J. Qiao, S. Zhi, D. Hu und W. Jiang, „RADRadar: Range-Angle-Doppler Feature Cube Reconstruction for Radar Scene Perception,“ *IEEE Robotics and Automation Letters*, Jg. 10, Nr. 7, S. 7278–7285, 2025. DOI: 10.1109/LRA.2025.3575312.

[52] M. Y. Lee, C. D. W. Lee, L. Shen und M. H. Ang, „RPC-Pillars: Radar Point Correction with Radar-PointPillars,“ in *Intelligent Autonomous Systems 18*, S.-G. Lee, J. An, N. Y. Chong, M. Strand und J. H. Kim, Hrsg., Cham: Springer Nature Switzerland, 2024, S. 573–585, ISBN: 978-3-031-44851-5.

[53] M. Akanksha, „A Comprehensive Review of Artificial Intelligence and Machine Learning: Concepts, Trends, and Applications,“ *International Journal of Scientific Research in Science and Technology*, Jg. 11, Nr. 5, S. 126–142, 2024. DOI: 10.32628/IJSRST2411587.

[54] M. I. Jordan und T. M. Mitchell, „Machine learning: Trends, perspectives, and prospects,“ *Science*, Jg. 349, Nr. 6245, S. 255–260, 2015. DOI: 10.1126/science.aaa8415.

[55] R. Gentleman und V. J. Carey, „Unsupervised Machine Learning,“ in *Bioconductor Case Studies*. New York, NY: Springer New York, 2008, S. 137–157. DOI: 10.1007/978-0-387-77240-0_10.

[56] R. S. Sutton und A. G. Barto, *Reinforcement Learning: An Introduction*, 2. Aufl. Cambridge, MA: MIT Press, 2018, ISBN: 978-0262039246.

[57] A. Vaswani u. a., „Attention is All You Need,“ in *Advances in Neural Information Processing Systems (NeurIPS)*, Bd. 30, 2017, S. 5998–6008.

[58] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg und D. Amodei, „Deep Reinforcement Learning from Human Preferences,“ *arXiv preprint arXiv:1706.03741*, 2017. Adresse: https://arxiv.org/abs/1706.03741.

[59] L. Ouyang u. a., „Training language models to follow instructions with human feedback,“ in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho und A. Oh, Hrsg., Bd. 35, Curran Associates, Inc., 2022, S. 27 730–27 744. Adresse: https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

[60] T. Jebara, *Machine learning: discriminative and generative*. Springer Science & Business Media, 2012, Bd. 755.

[61] Z. Chen, H. Sun, L. Zhang und F. Zhang, „Survey on Visual Signal Coding and Processing With Generative Models: Technologies, Standards, and Optimization,“ *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, Jg. 14, Nr. 2, S. 149–171, 2024. DOI: 10.1109/JETCAS.2024.3403524.

[62] S. Bond-Taylor, A. Leach, Y. Long und C. G. Willcocks, „Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models,“ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jg. 44, Nr. 11, S. 7327–7347, 2022. DOI: 10.1109/TPAMI.2021.3116668.

[63] S. Yazdani u. a., „Generative AI in depth: A survey of recent advances, model variants, and real-world applications,“ *Journal of Big Data*, Jg. 12, Nr. 1, S. 230, 2025. DOI: 10.1186/s40537-025-01247-x.

[64] J. Xiong u. a., „Autoregressive Models in Vision: A Survey,“ *Transactions on Machine Learning Research*, 2025, Survey Certification, ISSN: 2835-8856. Adresse: https://openreview.net/forum?id=1BqXkjNEGP.

[65] P. Kumar, „Diffusion Models and Generative Artificial Intelligence: Frameworks, Applications and Challenges,“ *Archives of Computational Methods in Engineering*, Jg. 32, Nr. 7, S. 4049–4092, Okt. 2025, ISSN: 1886-1784. DOI: 10.1007/s11831-025-10266-z.

[66] D. Carbone, „Hitchhiker’s guide on the relation of Energy-Based Models with other generative models, sampling and statistical physics: a comprehensive review,“ *Transactions on Machine Learning Research*, 2025, ISSN: 2835-8856. Adresse: https://openreview.net/forum?id=VTgixSbrJI.

[67] F.-A. Croitoru, V. Hondru, R. T. Ionescu und M. Shah, „Diffusion Models in Vision: A Survey,“ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jg. 45, Nr. 9, S. 10 850–10 869, 2023. DOI: 10.1109/TPAMI.2023.3261988.

[68] X. Hu u. a., „Diffusion Model for Image Generation - A Survey,“ in *2023 2nd International Conference on Artificial Intelligence, Human-Computer Interaction and Robotics (AIHCIR)*, 2023, S. 416–424. DOI: 10.1109/AIHCIR61661.2023.00073.

[69] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan und S. Ganguli, „Deep Unsupervised Learning using Nonequilibrium Thermodynamics,“ in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach und D. Blei, Hrsg., Ser. Proceedings of Machine Learning Research, Bd. 37, Lille, France: PMLR, Juli 2015, S. 2256–2265. Adresse: https://proceedings.mlr.press/v37/sohl-dickstein15.html.

[70] M. Kamiński und R. L. Ossowski, „Application of Shannon Entropy to Reaction–Diffusion Problems Using the Stochastic Finite Difference Method,“ *Entropy*, Jg. 27, Nr. 7, 2025. DOI: 10.3390/e27070705.

[71] D. Kingma und R. Gao, „Understanding diffusion objectives as the ELBO with simple data augmentation,“ *Advances in Neural Information Processing Systems*, Jg. 36, S. 65 484–65 516, 2023.

[72] D. Kingma, T. Salimans, B. Poole und J. Ho, „Variational Diffusion Models,“ in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang und J. W. Vaughan, Hrsg., Bd. 34, Curran Associates, Inc., 2021, S. 21 696–21 707. Adresse: https://proceedings.neurips.cc/paper_files/paper/2021/file/b578f2a52a0229873fefc2a4b06377fa-Paper.pdf.

[73] L. Lin, Z. Li, R. Li, X. Li und J. Gao, „Diffusion models for time-series applications: a survey,“ *Frontiers of Information Technology & Electronic Engineering*, Jg. 25, Nr. 1, S. 19–41, 2024. DOI: 10.1631/FITEE.2300310.

[74] C. P. Robert und G. Casella, *Monte Carlo Statistical Methods* (Springer Texts in Statistics), 2. Aufl. Springer New York, 2004. DOI: 10.1007/978-1-4757-4145-2.

[75] T. Li und K. He, *Back to Basics: Let Denoising Generative Models Denoise*, 2025. arXiv: 2511.13720 [cs.CV]. Adresse: https://arxiv.org/abs/2511.13720.

[76] 3Blue1Brown und W. Labs. „But how do AI images and videos actually work?“ Guest video by Welch Labs. (Juli 2025), Adresse: https://www.youtube.com/watch?v=iv-5mZ_9CPY.

[77] T. Salimans und J. Ho, „Progressive Distillation for Fast Sampling of Diffusion Models,“ in *International Conference on Learning Representations*, 2022. Adresse: https://openreview.net/forum?id=TIdIXIpzhoI.

[78] O. Chapelle, B. Schölkopf und A. Zien, Hrsg., *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006, ISBN: 978-0-262-51412-5.

[79] E. Facco, M. d’Errico, A. Rodriguez und A. Laio, „Estimating the intrinsic dimension of datasets by a minimal neighborhood information,“ *Scientific Reports*, Jg. 7, Nr. 1, S. 12 140, 2017. DOI: 10.1038/s41598-017-11873-y.

[80] I. Goodfellow, Y. Bengio und A. Courville, „Representation Learning,“ in *Deep Learning*, MIT Press, 2016, Kap. 5, S. 119–166.

[81] L. Liu, Y. Ren, Z. Lin und Z. Zhao, „Pseudo Numerical Methods for Diffusion Models on Manifolds,“ in *International Conference on Learning Representations*, 2022. Adresse: https://openreview.net/forum?id=PlKWVd2yBkY.

[82] O. Ronneberger, P. Fischer und T. Brox, „U-Net: Convolutional Networks for Biomedical Image Segmentation,“ in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells und A. F. Frangi, Hrsg., Cham: Springer International Publishing, 2015, S. 234–241.

[83] W. Peebles und S. Xie, „Scalable diffusion models with transformers,“ in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, S. 4195–4205.

[84] A. Q. Nichol und P. Dhariwal, „Improved Denoising Diffusion Probabilistic Models,“ in *Proceedings of the 38th International Conference on Machine Learning*, Ser. ICML, Bd. 139, PMLR, 2021, S. 8162–8171. Adresse: https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf.

[85] R. Rombach, A. Blattmann, D. Lorenz, P. Esser und B. Ommer, „High-Resolution Image Synthesis with Latent Diffusion Models,“ in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, S. 10 674–10 685. DOI: 10.1109/CVPR52688.2022.01042.

[86] K. He, X. Zhang, S. Ren und J. Sun, „Deep Residual Learning for Image Recognition,“ in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, S. 770–778. DOI: 10.1109/CVPR.2016.90.

[87] X. Wang, R. Girshick, A. Gupta und K. He, „Non-local Neural Networks,“ in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, S. 7794–7803.

[88] T. Karras, M. Aittala, J. Lehtinen, J. Hellsten, T. Aila und S. Laine, „Analyzing and Improving the Training Dynamics of Diffusion Models,“ in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, S. 24 174–24 184. DOI: 10.1109/CVPR52733.2024.02282.

[89] D. Morales-Brotons, T. Vogels und H. Hendrikx, „Exponential Moving Average of Weights in Deep Learning: Dynamics and Benefits,“ *Transactions on Machine Learning Research*, 2024, ISSN: 2835-8856. Adresse: https://openreview.net/forum?id=2M9CUnYnBA.

[90] C. Clason, „Metrische Räume,“ in *Einführung in die Funktionalanalysis*. Cham: Springer Nature Switzerland, 2025, S. 3–9. DOI: 10.1007/978-3-031-74714-4_1.

[91] V. Borisov, K. Sessler, T. Leemann, M. Pawelczyk und G. Kasneci, „Language Models are Realistic Tabular Data Generators,“ in *The Eleventh International Conference on Learning Representations*, 2023. Adresse: https://openreview.net/forum?id=cEygmQNOeI.

[92] G. Kalaimani, G. Kavitha, S. Chinnaiyan und S. Mylapalli, „Optimally configured generative adversarial networks to distinguish real and AI-generated human faces,“ *Signal, Image and Video Processing*, Jg. 18, Nr. 11, S. 7921–7938, 2024. DOI: 10.1007/s11760-024-03440-6.

[93] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet und S. Gelly, „Assessing Generative Models via Precision and Recall,“ in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi und R. Garnett, Hrsg., Bd. 31, Curran Associates, Inc., 2018.

[94] C. Szegedy u. a., „Going deeper with convolutions,“ in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, S. 1–9. DOI: 10.1109/CVPR.2015.7298594.

[95] J. A. O’Reilly und F. Asadi, „Pre-trained vs. Random Weights for Calculating Fréchet Inception Distance in Medical Imaging,“ in *2021 13th Biomedical Engineering International Conference (BMEiCON)*, 2021, S. 1–4. DOI: 10.1109/BMEiCON53485.2021.9745214.

[96] R. N. Padilla und V. Kreinovich, „Why Moments (and Generalized Moments) Are Used in Statistics and Why Expected Utility Is Used in Decision Making: A Possible Explanation,“ in *Decision Making Under Uncertainty and Constraints: A Why-Book*, M. Ceberio und V. Kreinovich, Hrsg. Cham: Springer International Publishing, 2023, S. 181–187, ISBN: 978-3-031-16415-6. DOI: 10.1007/978-3-031-16415-6_27.

[97] I. Grabec und W. Sachse, „Maximum Entropy Principles,“ in *Synergetics of Measurement, Prediction and Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, S. 137–165. DOI: 10.1007/978-3-642-60336-5_6.

[98] D. C. Dowson und B. V. Landau, „The Fréchet distance between multivariate normal distributions,“ *Journal of Multivariate Analysis*, Jg. 12, Nr. 3, S. 450–455, 1982. Adresse: https://EconPapers.repec.org/RePEc:eee:jmvana:v:12:y:1982:i:3:p:450-455.

[99] A. Obukhov und M. Krasnyanskiy, „Quality Assessment Method for GAN Based on Modified Metrics Inception Score and Fréchet Inception Distance,“ in *Software Engineering Perspectives in Intelligent Systems*, R. Silhavy, P. Silhavy und Z. Prokopova, Hrsg., Cham: Springer International Publishing, 2020, S. 102–114.

[100] W. B. Johnson und J. Lindenstrauss, „Extensions of Lipschitz mappings into a Hilbert space,“ in *Conference in Modern Analysis and Probability (New Haven, Conn., 1982)*, Ser. Contemporary Mathematics, R. Beals, A. Beck, A. Bellow u. a., Hrsg., Bd. 26, Providence, RI: American Mathematical Society, 1984, S. 189–206.

[101] D. Donoho, „Compressed sensing,“ *IEEE Transactions on Information Theory*, Jg. 52, Nr. 4, S. 1289–1306, 2006. DOI: 10.1109/TIT.2006.871582.

[102] E. J. Candes und T. Tao, „Near-Optimal Signal Recovery From Random Projections: Universal Encoding Strategies?“ *IEEE Transactions on Information Theory*, Jg. 52, Nr. 12, S. 5406–5425, 2006. DOI: 10.1109/TIT.2006.885507.

[103] E. Candes und T. Tao, „Decoding by linear programming,“ *IEEE Transactions on Information Theory*, Jg. 51, Nr. 12, S. 4203–4215, 2005. DOI: 10.1109/TIT.2005.858979.

[104] M. A. Davenport, M. F. Duarte, Y. C. Eldar und G. Kutyniok, „Introduction to Compressed Sensing,“ in *Compressed Sensing: Theory and Applications*, Y. C. Eldar und G. Kutyniok, Hrsg., Cambridge University Press, 2012, S. 1–64. DOI: 10.1017/CBO9780511794308.002.

[105] T. T. Cai, G. Xu und J. Zhang, „On Recovery of Sparse Signals Via ℓ_1 Minimization,“ *IEEE Transactions on Information Theory*, Jg. 55, Nr. 7, S. 3388–3397, 2009. DOI: 10.1109/TIT.2009.2021377.

[106] E. J. Candès, „The Restricted Isometry Property and Its Implications for Compressed Sensing,“ *Comptes Rendus Mathematique*, Jg. 346, Nr. 9-10, S. 589–592, 2008. DOI: 10.1016/j.crma.2008.03.014.

[107] A. Tillmann, „COAP 2019 Best Paper Prize: Paper of Andreas Tillmann,“ *Computational Optimization and Applications*, Jg. 77, Nr. 3, S. 623–626, 2020. DOI: 10.1007/s10589-020-00235-6.

[108] A. M. Tillmann und M. E. Pfetsch, „The Computational Complexity of the Restricted Isometry Property, the Nullspace Property, and Related Concepts in Compressed Sensing,“ *IEEE Transactions on Information Theory*, Jg. 60, Nr. 2, S. 1248–1259, 2014. DOI: 10.1109/TIT.2013.2290112.

[109] S. Foucart und H. Rauhut, „Sparse Recovery with Random Matrices,“ in *A Mathematical Introduction to Compressive Sensing*. New York, NY: Springer New York, 2013, S. 271–310. DOI: 10.1007/978-0-8176-4948-7_9.

[110] R. Vershynin, „Introduction to the non-asymptotic analysis of random matrices,“ in *Compressed Sensing: Theory and Applications*, Y. C. Eldar und G. Kutyniok, Hrsg. Cambridge University Press, 2012, S. 210–268. DOI: 10.1017/CBO9780511794308.002.

[111] G. Li, X. Xu und Y. Gu, „Lower Bound for RIP Constants and Concentration of Sum of Top Order Statistics,“ *IEEE Transactions on Signal Processing*, Jg. 68, S. 3169–3178, 2020. DOI: 10.1109/TSP.2020.2985848.

[112] M. Fornasier und H. Rauhut, „Compressive Sensing,“ in *Handbook of Mathematical Methods in Imaging*, O. Scherzer, Hrsg. New York, NY: Springer New York, 2015, S. 205–256, ISBN: 978-1-4939-0790-8. DOI: 10.1007/978-1-4939-0790-8_6.

[113] W. K. Härdle, L. Simar und M. R. Fengler, „Moving to Higher Dimensions,“ in *Applied Multivariate Statistical Analysis*. Cham: Springer International Publishing, 2024, S. 71–106. DOI: 10.1007/978-3-031-63833-6_3.

[114] T. F. Chan, G. H. Golub und R. J. Leveque, „Algorithms for Computing the Sample Variance: Analysis and Recommendations,“ *The American Statistician*, Jg. 37, Nr. 3, S. 242–247, 1983. DOI: 10.1080/00031305.1983.10483115.

[115] B. P. Welford, „Note on a Method for Calculating Corrected Sums of Squares and Products,“ *Technometrics*, Jg. 4, Nr. 3, S. 419–420, 1962. DOI: 10.1080/00401706.1962.10490022.

[116] A. Borodin und R. El-Yaniv, *Online computation and competitive analysis*. USA: Cambridge University Press, 1998, ISBN: 0521563925.

[117] S. Albers, „Online algorithms: a survey,“ *Mathematical Programming*, Jg. 97, Nr. 1, S. 3–26, 2003. DOI: 10.1007/s10107-003-0436-0.

[118] S. Bischoff u. a., „A Practical Guide to Sample-based Statistical Distances for Evaluating Generative Models in Science,“ *Transactions on Machine Learning Research*, 2024, ISSN: 2835-8856. Adresse: https://openreview.net/forum?id=isEFziui9p.

[119] D. N. Ayyala, S. Ghosh und D. F. Linder, „Covariance matrix testing in high dimension using random projections,“ *Computational Statistics*, Jg. 37, Nr. 3, S. 1111–1141, 2022. DOI: 10.1007/s00180-021-01166-4.

[120] O. Ledoit und M. Wolf, „A well-conditioned estimator for large-dimensional covariance matrices,“ *Journal of Multivariate Analysis*, Jg. 88, Nr. 2, S. 365–411, 2004. DOI: 10.1016/S0047-259X(03)00096-4.

[121] J. Fan, Y. Liao und H. Liu, „An overview of the estimation of large covariance and precision matrices,“ *The Econometrics Journal*, Jg. 19, Nr. 1, S. C1–C32, März 2016. DOI: 10.1111/ectj.12061.

[122] D. Gusland, J. M. Christiansen, B. Torvik, F. Fioranelli, S. Z. Gurbuz und M. Ritchie, „Open Radar Initiative: Large Scale Dataset for Benchmarking of micro-Doppler Recognition Algorithms,“ in *2021 IEEE Radar Conference (RadarConf21)*, 2021, S. 1–6. DOI: 10.1109/RadarConf2147009.2021.9455239.

[123] R. E. Burkard und E. Çela, „Linear Assignment Problems and Extensions,“ in *Handbook of Combinatorial Optimization: Supplement Volume A*, D.-Z. Du und P. M. Pardalos, Hrsg. Boston, MA: Springer US, 1999, S. 75–149. DOI: 10.1007/978-1-4757-3023-4_2.

[124] T. Takiguchi, J. Bilmes, M. Yoshii und Y. Ariki, „Evaluation of random-projection-based feature combination on speech recognition,“ in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010, S. 2150–2153. DOI: 10.1109/ICASSP.2010.5495595.

[125] Y. Kim und K.-A. Toh, „Sparse random projection for efficient cancelable face feature extraction,“ in *2008 3rd IEEE Conference on Industrial Electronics and Applications*, 2008, S. 2139–2144. DOI: 10.1109/ICIEA.2008.4582897.

[126] G. Huang u. a., *An empirical study on evaluation metrics of generative adversarial networks*, 2018. Adresse: https://openreview.net/forum?id=Sy1f0e-R-.

[127] J. Monsalve, J. Ramirez, I. Esnaola und H. Arguello, „Covariance Estimation From Compressive Data Partitions Using a Projected Gradient-Based Algorithm,“ *IEEE Transactions on Image Processing*, Jg. 31, S. 4817–4827, 2022. DOI: 10.1109/TIP.2022.3187285.

[128] R. Fernandes de Mello und M. Antonelli Ponti, „Statistical Learning Theory,“ in *Machine Learning: A Practical Approach on the Statistical Learning Theory*. Cham: Springer International Publishing, 2018, S. 75–128. DOI: 10.1007/978-3-319-94989-5_2.

[129] A. Cennamo, F. Kaestner und A. Kummert, „Leveraging Radar Features to Improve Point Clouds Segmentation with Neural Networks,“ in *Proceedings of the 21st EANN (Engineering Applications of Neural Networks) 2020 Conference*, L. Iliadis, P. P. Angelov, C. Jayne und E. Pimenidis, Hrsg., Cham: Springer International Publishing, 2020, S. 119–131, ISBN: 978-3-030-48791-1.

[130] Y. Zhu, Z. Li, T. Wang, M. He und C. Yao, „Conditional Text Image Generation with Diffusion Models,“ in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Open Access version provided by the Computer Vision Foundation; identical to the accepted version except for watermark, 2023. Adresse: https://openaccess.thecvf.com/content/CVPR2023/papers/Zhu_Conditional_Text_Image_Generation_With_Diffusion_Models_CVPR_2023_paper.pdf.

[131] K. Mei, M. Delbracio, H. Talebi, Z. Tu, V. M. Patel und P. Milanfar, „CoDi: Conditional Diffusion Distillation for Higher-Fidelity and Faster Image Generation,“ in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, S. 9048–9058. DOI: 10.1109/CVPR52733.2024.00864.

[132] L. Veeramacheneni, M. Wolter, H. Kuehne und J. Gall, „Fréchet Wavelet Distance: A Domain-Agnostic Metric for Image Generation,“ in *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*, OpenReview.net, 2025.

A. Evaluierung der Log-Likelihood-Schätzung für Pseudoradar-Generatoren

In diesem Abschnitt wird die Zuverlässigkeit sowie das Verhalten der in Abschnitt 3.2.1 beschriebenen Log-Likelihood-Schätzer für synthetisch generierte Pseudoradar-Punktwolken untersucht (siehe Kapitel 3.2). Ziel der Analyse ist es, die durch den Detektionsalgorithmus (Algorithmus 4) geschätzten Parameter (Anzahl der Linien, Anzahl der Punkte pro Linie sowie Anzahl der Clutter-Punkte) mit den tatsächlichen, bei der Generierung gezogenen Parametern zu vergleichen. Auf diese Weise lässt sich beurteilen, wie zuverlässig die Log-Likelihood-Schätzung auf Basis der detektierten Parameter ist. Die Evaluierung erfolgt unter Verwendung eines Pseudoradar-Generators, der für jede Punktwolke folgende Poisson-Parameter nutzt:

$$\lambda_{\text{Linien}} = 5, \lambda_{\text{Punkte/Linie}} = 20, \lambda_{\text{Clutter}} = 50.$$

Für jede erzeugte Punktwolke werden die aus den Poisson-Verteilungen gezogenen wahren Parameter gespeichert. Anschließend werden die Parameter derselben Punktwolken mittels Algorithmus 4 geschätzt. Abbildung A.1 zeigt exemplarisch drei generierte Punktwolken sowie die darin detektierten Linien. Die resultierenden Verteilungen der wahren und detektierten Parameter für 1000 generierte Samples sind in Abbildung A.2 dargestellt.

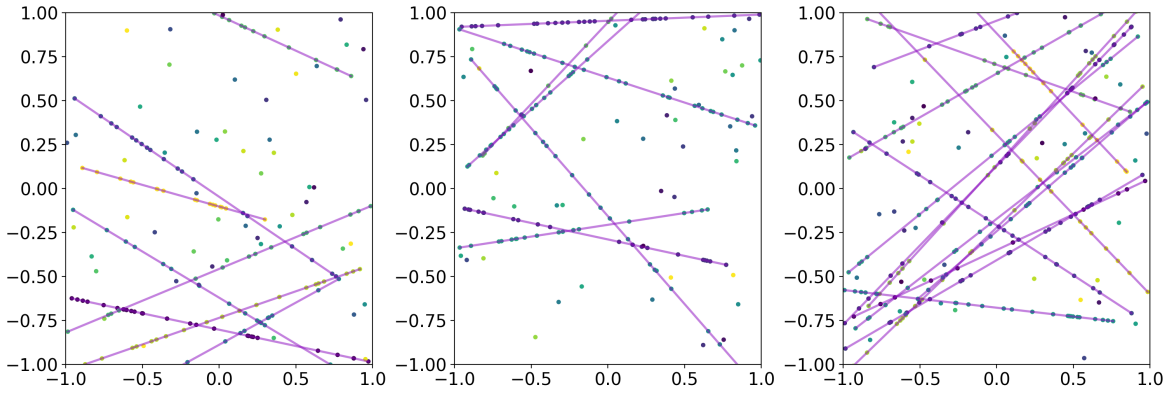

Abbildung A.1.: Drei generierte Pseudoradar-Punktwolken, in denen die durch Algorithmus 4 detektierten Linien eingezeichnet sind.

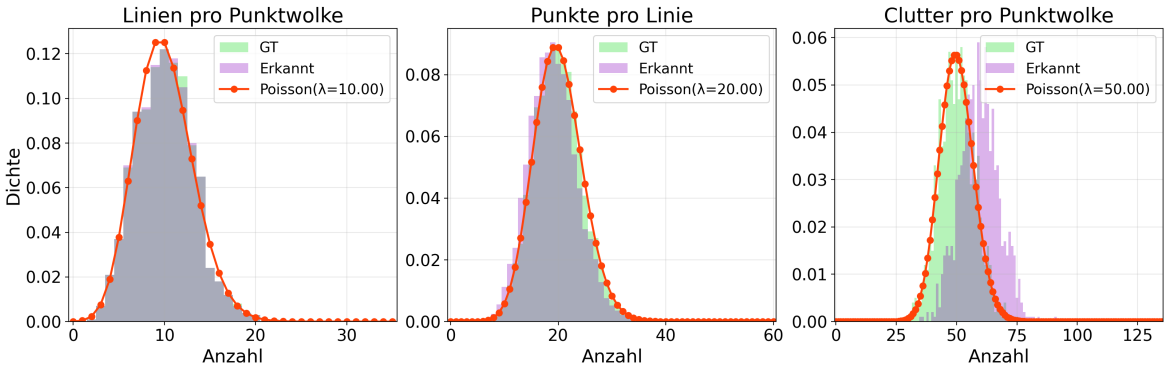

Abbildung A.2.: Verteilung der wahren und detektierten Anzahl an Linien, Punkten pro Linie und Clutter-Punkten für 1000 generierte Punktwolken und der wahren Poisson-Verteilung.

Tabelle A.1.: Log-Likelihood Ergebnisse für 1000 generierte Punktwolken, die die Log-Likelihood auf Basis detektierter und wahrer Parameter vergleicht.

Metrik	**Mittlere Log-Likelihood**
Geschätzte Parameterwerte	-9.798538
Wahre Parameterwerte	-8.859375
Erwartungswert (Theorie)	-8.847222

Die Ergebnisse verdeutlichen, dass die Anzahl der Linien zuverlässig geschätzt wird, während die Anzahl der Punkte pro Linie systematisch unterschätzt wird. Dadurch ergibt sich eine insgesamt zu geringe Anzahl detektierter Linienpunkte, was gleichzeitig zu einer Überschätzung der Clutter-Punkte führt. Eine mögliche Ursache hierfür liegt in der heuristischen Punktezuordnung im Detektionsverfahren. Da Punkte sukzessive Linien zugeordnet und anschließend aus der Menge der verfügbaren Punkte entfernt werden, kann die Auswahl eines nahegelegenen Clutter-Punkts im Entscheidungsprozess dazu führen, dass nur Teile der Punkte auf einer Linie erkannt werden. In der Folge verbleiben zugehörige Linienpunkte im Datensatz und werden fälschlicherweise als Clutter klassifiziert. Dieses Verhalten wirkt sich unmittelbar auf die Log-Likelihood-Schätzung aus. Die in Tabelle A.1 dargestellten mittleren Log-Likelihood-Werte über alle 1000 Samples zeigen eine klare Abweichung zwischen der Log-Likelihood, die anhand der detektierten Parameter berechnet wurde, und denen, die auf Basis der tatsächlich gezogenen Parameter berechnet wurden. Dies bestätigt, dass der Detektionsalgorithmus die zugrundeliegenden Modellparameter nicht exakt rekonstruieren kann, was die Genauigkeit der Log-Likelihood-Schätzung beeinflusst. Trotz dieser systematischen Abweichungen bleibt die Log-Likelihood-Schätzung jedoch sinnvoll und praktikabel. Die

Verzerrungen treten über alle generierten Punktwolken konsistent auf und führen somit zu einer stabilen sowie reproduzierbaren Abweichungsstruktur. Dadurch reflektieren die geschätzten Log-Likelihood-Werte das relative Verhalten verschiedener Generatorvarianten zuverlässig, selbst wenn die absoluten Werte nicht exakt den echten Werten entsprechen. Insbesondere im Rahmen vergleichender Evaluierungen, wie etwa beim Benchmarking verschiedener Generatorarchitekturen, stellt die auf den geschätzten Parametern basierende Log-Likelihood daher weiterhin eine robuste und für vergleichende Analysen geeignete Bewertungsgröße dar.

B. Zusätzliche Ergebnisse zur Validierung der FRD (RQ1)

B.1. Fréchet-Radar-Distance (FRD) bei gleichen Pseudoradar-Generator-Parametern

Im Folgenden werden zusätzlich zu den in Abschnitt 4.1.1 vorgestellten Ergebnissen weitere Pseudoradar-Generatoren auf die Konvergenz ihrer Ergebnisse gegen Null überprüft. Die Abbildungen zeigt im ersten Plot von links eine diskretisierte Realisierung des in diesem Experiment verwendeten Pseudoradar-Generators. Im zweiten Plot wird die FRD für die verschiedenen Feature-Dimensionen dargestellt. Im dritten Plot wird die nach der Dimensionalität der zufälligen Projektionen unskalierte FRD dargestellt, um zu zeigen, dass der in Abschnitt 4.1.1 gefundene Zusammenhang auch für andere Pseudoradar-Generatoren gilt.

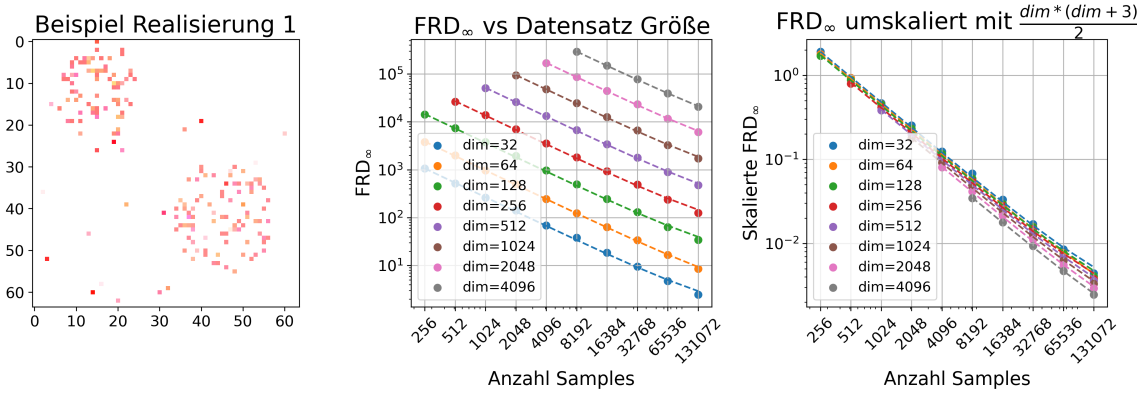

Abbildung B.1.: Experiment eines Pseudoradar-Generators, der Punkte, die sich in Kreisen anordnen, und Clutter-Punkte erzeugt.

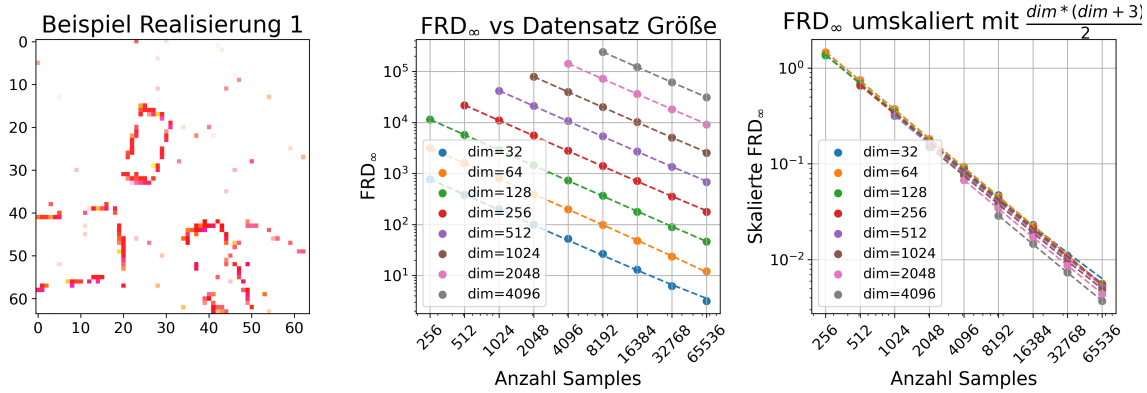

Abbildung B.2.: Experiment eines Pseudoradar-Generators, der Clutter-Punkte und Punkte, die sich auf dem Rand von Rechtecken anordnen, erzeugt.

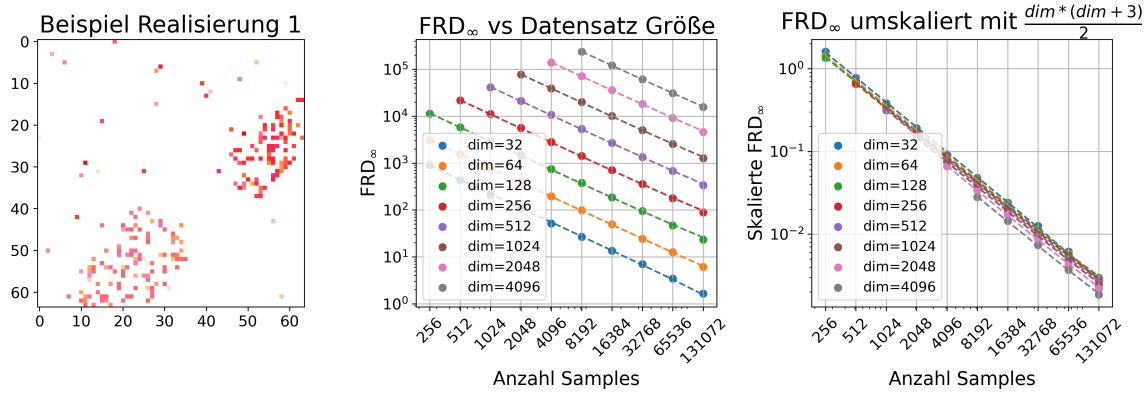

Abbildung B.3.: Experiment eines Pseudoradar-Generators, der Clutter-Punkte und Punkte, die sich auf der Fläche von Rechtecken anordnen, erzeugt.

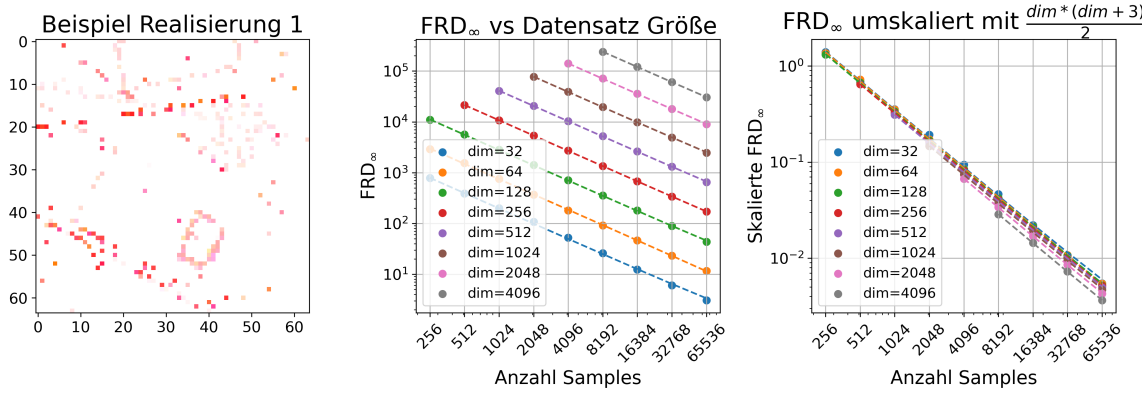

Abbildung B.4.: Experiment eines Pseudoradar-Generators, der Clutter-Punkte und eine Mischung aus Punkten, die sich innerhalb von Linien, Rechtecken oder Kreisen anordnen können, erzeugt.

B.2. FRD bei unterschiedlichen Pseudoradar-Generatoren mit Variation in nur einem Parameter

Im Folgenden werden ergänzend zu dem in Abschnitt 4.1.2.2 vorgestellten Experiment, bei dem die Anzahl der Linien variiert wurde, während die anderen Parameter konstant geblieben sind, weitere Experimente durchgeführt, in denen jeweils nur ein einzelner Parameter des Pseudoradar-Generators variiert wird. Die Abbildungen zeigen jeweils die FRD, ausgewertet mit einer Feature-Dimension von 4096. In Abbildung B.6 wird nur die Anzahl der Punkte pro Linie variiert, während in Abbildung B.5 nur die Anzahl der Clutter-Punkte verändert wird. Die Resultate unterscheiden sich dabei nicht von dem Experiment, in dem die Anzahl der Linien variiert wurde. Abbildung B.5 weist zwar insgesamt niedrigeren Werte der FRD und der approximierten Log-Likelihood auf, sie sind jedoch darauf zurückzuführen, dass die Anzahl der Clutter-Punkte im Vergleich zu den Punkten, die sich entlang der Linien anordnen, geringer ist. Dadurch ändert sich die Struktur des generierten Radarsignals nur geringfügig.

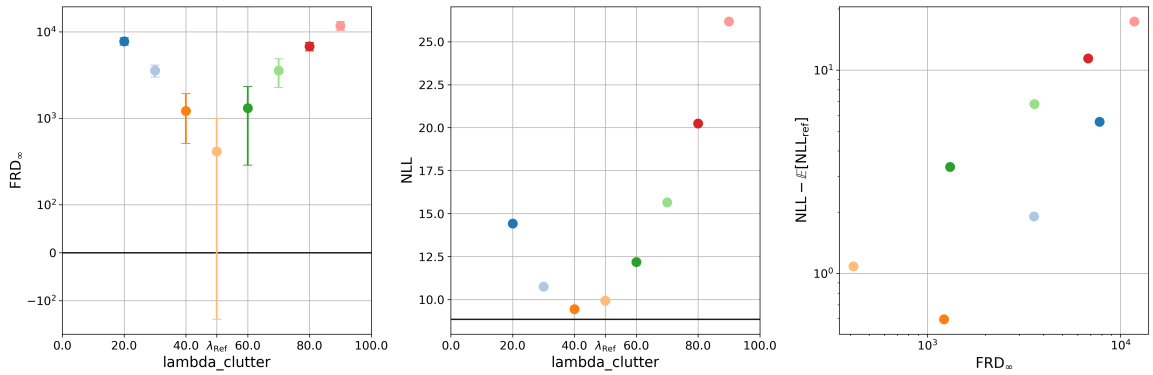

Abbildung B.5.: Visualisierung der FRD_∞ (Projektionsdimension 4096), der approximierten Log-Likelihood und der Abhängigkeit beider Größen voneinander für verschiedene Radar-Generatoren bei denen nur die Anzahl der Clutter-Punkte variiert. Die Anzahl der Linien und die Anzahl an Punkten pro Linie ist über alle Generatoren konstant und äquivalent zum Referenzgenerator mit $\lambda_{\text{Linien}} = 10, \lambda_{\text{Clutter}} = 50$.

B.3. FRD bei erweiterten Pseudoradar-Generatoren

In diesem Abschnitt wird ein Experiment vorgestellt, in dem die FRD für verschiedene erweiterte Pseudoradar-Generatoren untersucht wird. Die betrachteten Generatoren erzeugen Punktmuster, die jeweils entlang einer von

Rang	Kategorie	FRD_∞	$\Delta\lambda_{Lines}$	$\Delta\lambda_{PtsLine}$	$\Delta\lambda_{Circle}$	$\Delta\lambda_{PtsCircle}$	$\Delta\lambda_{Rect}$	$\Delta\lambda_{PtsRect}$	$\Delta\lambda_{RectOut}$	$\Delta\lambda_{PtsRectOut}$
1	lines	566.10	0	0	0	0	0	0	0	0
2	lines	1426.21	-2	+5	0	0	0	0	0	0
3	lines	4714.05	+10	-10	0	0	0	0	0	0
4	lines	6774.69	+15	-12	0	0	0	0	0	0
5	lines	12993.63	-5	+20	0	0	0	0	0	0
6	rect_outline	14911.00	-10	-20	0	0	0	0	+10	+20
7	circles	15982.36	-10	-20	+25	+8	0	0	0	0
8	circles	16021.50	-10	-20	+20	+10	0	0	0	0
9	rect_outline	16111.17	-10	-20	0	0	0	0	+20	+10
10	rect_outline	16480.00	-10	-20	0	0	0	0	+8	+25
11	rect_outline	17273.81	-10	-20	0	0	0	0	+25	+8
12	circles	22407.75	-10	-20	+10	+20	0	0	0	0
13	rect_outline	27544.42	-10	-20	0	0	0	0	+5	+40
14	circles	28530.36	-10	-20	+8	+25	0	0	0	0
15	rectangles	29616.94	-10	-20	0	0	+10	+20	0	0
16	rectangles	29889.28	-10	-20	0	0	+8	+25	0	0
17	rectangles	30988.13	-10	-20	0	0	+20	+10	0	0
18	rectangles	31428.71	-10	-20	0	0	+25	+8	0	0
19	rectangles	31980.99	-10	-20	0	0	+5	+40	0	0
20	circles	53584.07	-10	-20	+5	+40	0	0	0	0

Tabelle B.1.: Rangliste der Generatorparameter nach FRD_∞. Die FRD_∞ wurde mittels zufälliger Projektion mit Zieldimension 4096 berechnet. Dargestellt sind die Differenzen zwischen dem Referenzgenerator und den Vergleichsgeneratoren. Die Anzahl der Clutter-Punkte ist für alle Generatoren identisch und wird daher nicht aufgeführt.

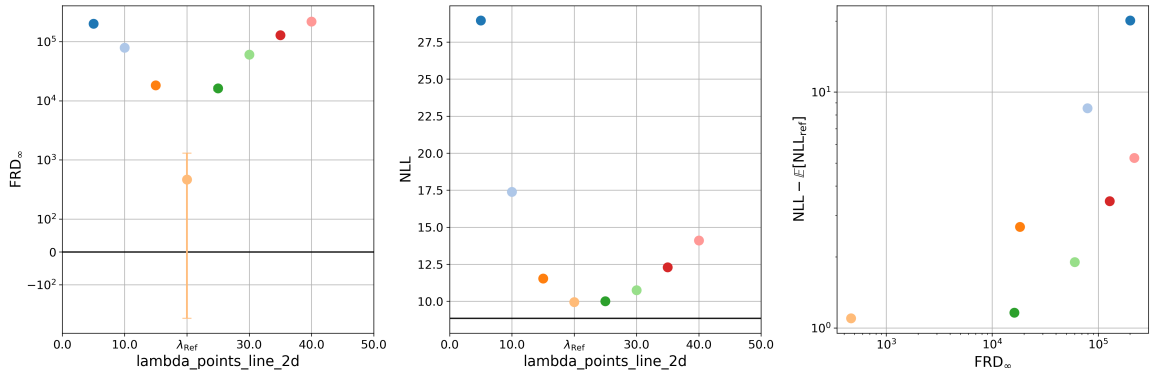

Abbildung B.6.: Visualisierung der FRD_∞, der approximierten Log-Likelihood und der Abhängigkeit beider Größen voneinander für verschiedene Radar-Generatoren bei denen nur die Anzahl der Punkte pro Linie variiert. Die Anzahl der Linien und die Anzahl an Clutter-Punkten ist über alle Generatoren konstant und äquivalent zum Referenzgenerator mit $\lambda_{\text{Linien}} = 10, \lambda_{\text{Punkte/Linie}} = 20$.

Kategorie	Durchschnittlicher Rang	Durchschnittliche FRD_∞
lines	3.0	5294
rect_outline	9.8	18464.
circles	12.2	27305
rectangles	17.0	30780

Tabelle B.2.: Durchschnittliche Gesamtplatzierung und FRD_∞ pro Kategorie.

vier geometrischen Strukturen angeordnet sind: (1) Linien, (2) Kreisflächen, (3) Flächen von Rechtecken oder (4) Außenkanten von Rechtecken (vgl. Abschnitt B.1). Jeder Generator ist dabei einer eindeutigen Kategorie zugeordnet. Alle Generatoren erzeugen die gleiche Anzahl an Clutter-Punkten sowie die gleiche Anzahl an erwarteten Punkten. Da sich die erzeugten Punkte auf unterschiedlichen geometrischen Strukturen befinden, kann im Gegensatz zu Abschnitt 4.1.2.2 keine Log-Likelihood als Referenzmetrik bestimmt werden. Die Ergebnisse sind in Tabelle B.1 dargestellt. Die Bewertung der Ähnlichkeit basiert ausschließlich auf der jeweiligen Strukturkategorie. Tabelle B.2 zeigt daher zusätzlich den durchschnittlichen Rang und die durchschnittliche FRD_{∞} pro Kategorie. Die Ergebnisse sind insofern konsistent, als, dass die Generatoren, die Punkte entlang von Linien erzeugen, durchgehend die besten Platzierungen erzielen. Der zweite Platz der Generatoren, die Punkte auf den Außenkanten von Rechtecken erzeugen, lässt sich plausibel damit erklären, dass diese Struktur vollständig aus vier linearen Segmenten besteht, die sich jeweils im 90°-Winkel schneiden. Dadurch weisen sie eine stärkere strukturelle Ähnlichkeit zu Liniengeneratoren auf als zu Kreisflächen oder Flächen von Rechtecken. Der Unterschied zwischen Kreisen und Rechtecken lässt sich jedoch nicht eindeutig erklären. Eine mögliche Hypothese wäre, dass Rechtecke aufgrund ihrer linearen Segmente eine geringere Distanz aufweisen müssten. Diese Vermutung wird jedoch durch die Messergebnisse widerlegt. Insgesamt zeigen die Experimente, dass die erwarteten Ähnlichkeitsbewertungen konsistent bleiben, selbst wenn die Generatoren Punkte entlang unterschiedlicher geometrischer Strukturen erzeugen.