Synthese und Analyse elektronenmikroskopischer Aufnahmen mittels künstlicher Intelligenz

Dissertation

Zur Erlangung des akademischen Grades eines

Doktor der Naturwissenschaften

Doctor Rerum Naturalium

vorgelegt von

Jonas Bals

Fakultät für Chemie der Universität Duisburg-Essen

Anorganische Chemie

DOI: 10.17185/duepublico/78379
URN: urn:nbn:de:hbz:465-20230511-082527-6

Die vorliegende Dissertation wurde im Zeitraum von Mai 2020 bis Dezember 2022 am Institut für Anorganische Chemie der Universität Duisburg-Essen unter Anleitung von Prof. Dr. Matthias Epple angefertigt.

1. Gutachter:	Prof. Dr. Matthias Epple
2. Gutachter:	Jun.-Prof. Dr. Kai S. Exner
Vorsitzender:	Jun.-Prof. Sven Heiles
Tag der Disputation:	27.04.2023

Memento Mori…

Inhaltsverzeichnis

Definitionen

Annotation	Bild, in dem jedes Pixel von einem Menschen gelabelt wurde. D. h. jedes Pixel ist einer Klasse zugeordnet. Die Klassen sind typischerweise Vordergrund & Hintergrund. Die Anzahl der Klassen kann aber auch erweitert werden, um diverse Klassen zu erfassen.
Bild-Eigenschaften	Typische Merkmale, die das Bild ausmachen. Eigenschaften sind das Aussehen des Bildes. Einzelne Eigenschaften des Bildes sind von menschlicher Seite aus nicht zu erkennen. Beispielsweise sind alle Kanten, die von links nach rechts einen Intensitätswechsel begleiten, eine Eigenschaft.
Datensatz	Menge aus Einzeldaten (z. B. Bildern) oder korrelierten Einzeldaten-Paaren (Bild & Label, optional Gewichtung).
Trainings-Datensatz	Repräsentativer Anteil eines Datensatzes. Typischerweise 80 % aller Daten des Datensatzes. Wird in der Trainingsphase des Modells eingesetzt. Aus diesen Daten lernt das Modell.
Validierungs-Datensatz	Repräsentativer Anteil eines Datensatzes. Typischerweise 20 % aller Daten des Datensatzes. Diese Daten werden ausschließlich für das Testen des Modells genutzt.
Eigenschafts-Karte	Eine Matrix, die Bereiche des Auftretens einer Eigenschaft darstellt. Z. B. zeigen Bereiche der Eigenschaftskarte eines Kantenfilters alle Kanten einer spezifischen Richtung des Bildes auf.
Eigenschafts-Raum	Gesamtheit aller Eigenschaften einer Bildgruppe, wie z. B. einem Bilddatensatz.
Eigenschafts-Tensor	Ein Stapel von Eigenschafts-Karten, die in z-Richtung gestapelt wurden.
Hintergrund	Bereiche / Pixel des Bildes, die nicht für die Auswertung des Bildes von Interesse sind.
Hintergrundobjekt	Bereiche des Bildes, die von der KI als Objekt erkannt wurden, obwohl sie kein Objekt enthalten. Pixel dieser Objekte sind falsch positive (FP) Werte.
Label	Bezeichnung / Klassifizierung eines Datenpunktes. Das Modell lernt, die Wahrscheinlichkeit des wahren Labels vorherzusagen.

Objekt / Vordergrundobjekt	Der Bereich des Bildes, der für die Auswertung des Bildes von Interesse ist. Einzelne Vordergrundobjekte sind von einem Rand aus Hintergrund voneinander abgegrenzt.
Objekt-Eigenschaft	Messbare Größen eines Objektes. Z. B. Ausdehnung in x, y-Richtung, Fläche, Länge der Kontur, mittlere Pixelintensität der zum Objekt gehörenden Pixel, Rauschen, konvexe Hülle, Sphärizität.
Segmentierung	Visualisierung des argumentativen Maximums der Wahrscheinlichkeitsverteilung für jedes Pixel eines Bildes. Das Segmentierungsmodell lernt, aus Bild/Annotations-Paaren die menschliche Annotation nachzuahmen.

1 Einleitung und Zielsetzung der Arbeit

Produkte der heutigen Welt enthalten oftmals Mikro- und Nanopartikel. Die besonderen chemischen und physikalischen Eigenschaften von Nanopartikeln gegenüber ihren Bulkmaterialien machen diese Materialien zu einem aktuellen Forschungsthema. Synthetisch hergestellte Nanopartikel können über die Kontrolle der Syntheseparameter wünschenswerte Eigenschaften aufweisen. Zu diesen Eigenschaften gehören eine höhere elektrische Leitfähigkeit, eine antibakterielle Wirksamkeit, hohe chemische Reaktivität oder katalytische Wirkung. [1, 2] In der Technik können Nanopartikel dazu beitragen, leistungsstärkere und kleinere Computerbauteile zu entwickeln, während sie in der Medizin für den gerichteten Einsatz von Medikamenten im Körper genutzt werden können. [3, 4] Auch in Alltagsprodukten wie Deodorants oder Sonnencremes finden Nanopartikel als Biozide oder UV-Schutz Anwendung. In der Lebensmittelindustrie werden sie als Zusatzstoffe zur Farberhaltung oder als Trennmittel eingesetzt. Farbstoffe im Allgemeinen erhalten oft erst durch die Zugabe von spezifischen Nanopartikeln wünschenswerte Eigenschaften wie UV-Beständigkeit oder Leuchtkraft.

Für die wissenschaftliche Forschung sind Nanopartikel interessant, weil unterschiedliche Formen und Größen von Nanopartikeln in unterschiedlichen Anwendungsbereichen genutzt werden können. Nicht nur Größe und Form von Nanopartikeln entscheiden über ihre Anwendung, sondern auch das Material, aus dem diese hergestellt sind. Die Bandbreite der möglichen Materialkombinationen und Kristallstrukturen ist dabei nahezu unendlich. Da die Eigenschaften von Nanopartikeln von ihrer Form, Größe, Material und Kristallinität abhängen, müssen diese Faktoren nach der Synthese verifiziert werden, um eine sichere Nutzung der Partikel zu ermöglichen. Sollten diese Parameter von einem definierten Bereich abweichen, können spezifische Eigenschaften verschwinden, wodurch in dem Produkt unerwünschte Nebeneffekte auftreten können.

Für die Analyse der Größe werden typischerweise Methoden wie Scheibenzentrifugation (DCS) oder die dynamische Lichtstreuung (DLS) verwendet. Die Bestimmung von Größe und Form der Partikel ist mittels Rasterelektronen-Mikroskopie (REM) möglich. Das REM erzeugt hochauflösende Bilder der Probe, um Oberflächenstrukturen und die Dimension der Partikel abzubilden. Moderne REM Geräte können in 2 Modi betrieben werden. Zum einen ist die

Detektion von Sekundärelektronen (SE), zum anderen die Detektion von Transmissionselektronen (STEM) möglich. SE-Aufnahmen zeigen eine hohe Schärfentiefe, die es ermöglicht, die Oberfläche der Probe in einem pseudo-dreidimensionalen Bild über einen weiten Bereich abzubilden. STEM-Bilder wiederum verfügen über eine höhere räumliche Auflösung und zeigen zweidimensionale Schnitte der Probe. Beide Bildtypen unterscheiden sich in ihrem Aussehen stark voneinander.

Die Bestimmung der Partikelgröße und Form aus REM-Bildern bietet die Möglichkeit, die Partikel präzise zu charakterisieren. Um aus einem STEM- oder SE-Bild die mittlere Partikelgröße der Probe zu bestimmen, muss ein menschlicher Analyst mehrere Bilder einer Probe auswerten. Dieser Prozess ist nicht nur zeitaufwendig, sondern auch subjektiv und fehleranfällig.

Tiefe neuronale Netze (Deep *Neural Networks, DNNs*) und speziell faltende neuronale Netze (*Convolutional Neural Networks, CNNs*) sind probabilistische Modelle aus dem Bereich des maschinellen Lernens. Sie sind in der Lage, Bilder zu verarbeiten, indem sie spezifische Eigenschaften des Bildes identifizieren und basierend auf den Eigenschaften des Bildes eine Schätzung ausgeben, was in diesem Bild zu sehen ist (*Image Classification*), wo sich das Objekt befindet (*Object Localisation*) und welche Pixel zu diesem Objekt gehören (*Image Segmentation*). Somit bieten Methoden aus dem Bereich des maschinellen Lernens die Möglichkeit, die Analyse von Bildern zu automatisieren. Die Identifizierung von Objekten in licht- und elektronenmikroskopischen Aufnahmen mittels künstlicher Intelligenz (KI) und maschinellem Lernen (ML) ist in der Literatur bereits verbreitet. Etabliert sind bisher jedoch vor allem Methoden mittels manuell erstellter Eigenschaftsdetektoren. Manuelle Eigenschaftsdetektoren fassen Algorithmen zusammen, in welchen Objekte nach den vom Menschen bestimmten Eigenschaften identifiziert, vermessen und klassifiziert werden. Auch Methoden wie *Random Forest Classification* zur Segmentierung eines Bildes in Vordergrund (einzelne Nanopartikel) und Hintergrund (Probenträger) werden oftmals genutzt. [5, 6] Auch werden Bilder mittels adaptiver Schwellwertfindung segmentiert [7-9] oder nutzen von Menschen erstellte Filter zur Identifizierung von Partikeln. [10] Baiyasi et al. verfeinerten die so erhaltenen Konturen der Vordergrundobjekte, um mittels des *Watershed*-Algorithmus die Konturen der Objekte besser zu approximieren. [11] Diese Methoden haben den Vorteil, dass sie schnell zur Verfügung stehen und im Vergleich zu einem tiefen neuronalen Netzwerk eine geringere Parameteranzahl besitzen. Dadurch sind diese schneller verfügbar und können mit

weniger Daten trainiert werden. Der Nachteil ist jedoch, dass sie in komplexen Bildern versagen. Vordergrundobjekte können nicht korrekt aufgetrennt werden. Die präzise Segmentierung von Nanopartikeln ist mit dieser Methode nicht vollständig gegeben. [5, 11] Die Klassifizierung der segmentierten Partikel erfolgt beispielsweise mit einer Hauptkomponentenanalyse der Objekteigenschaften. [6, 12] Die Eigenschaften der Partikel werden von der Kontur sowie Fläche des Partikels abgeleitet (u. a. konvexe Hülle, Länge der Kontur, Sphärizität des Partikels). Durch die bereits schlechte Segmentierung einzelner Partikel ist es kaum möglich, diese präzise einer Morphologie zuzuordnen.

Modernere Methoden, die auf tiefem maschinellen Lernen basieren, wurden bisher selten genutzt. So zeigen Kim et al. zwar die Nutzung von eines *Faster Regionproposal Convolutional Neural Network* (*Faster RCNN*) [13, 14] für die Erkennung von Nanopartikeln auf, allerdings nur für die Identifikation von 4 verschiedenen Formen. [15] Ebenfalls zeigen Holm et al. eine Reihe von möglichen Anwendungsgebieten von tiefem Lernen in den Materialwissenschaften auf. Es wurde gezeigt, dass faltende neuronale Netze für die Erkennung von Strukturen und Formen in elektronenmikroskopischen Aufnahmen nutzbar sind. Ein konkretes Modell für die Erkennung von Nanopartikeln wurde nicht vorgestellt. [16, 17] Die Erkennung von Strukturen wie Nanopartikeln erfolgt dabei über etablierte Modelle zur Erkennung von Objekten wie VGG [18] oder ResNet. [19] Weiterhin wurden Methoden des automatisierten Clusterns mittels *k-means* Algorithmus für die Analyse der Diversität von Partikeln in elektronen-mikroskopischen Aufnahmen genutzt. [20]

Eine weitere Problematik für die Erkennung von diversen Nanopartikelformen ist der Mangel an definierten Datensätzen. So wurden bisher Datensätze zu allgemeinen Mikrostrukturen im elektronenmikroskopischen Kontext veröffentlicht. [15, 21] Die vorgestellten Datensätze zeigen jedoch entweder zu einseitige oder allgemeine Strukturen, um für das Training eines KI-Modells zur Erkennung von Nanopartikeln eingesetzt zu werden. Ein großer Datensatz für Nanopartikelformen ist derzeitig nicht verfügbar.

Für diese Arbeit wurden dementsprechend mehrere Datensätze für die Erkennung von Nanopartikeln in STEM- und SE-Bildern erarbeitet. Diese erlaubten es, tiefe neuronale Netzwerke zu trainieren, die in jüngerer Zeit zu einem Paradigmenwechsel der automatischen Bildverarbeitung geführt haben. [18, 19, 22-24] Das Training der Netzwerke erfolgte explizit für die Segmentierung und Klassifizierung von Nanopartikeln. Zwei getrennte Workflows wurden

ausgearbeitet. Zunächst wurden STEM-Bilder bearbeitet. Hierfür wurde die bestehende Methode zur Erkennung von Objektgrenzen mittels Gewichtungskarten weiter verbessert. Dadurch können mehr Objekte in STEM-Bildern sicher voneinander getrennt werden. Weiterhin wurden SE-Bilder bearbeitet. Pro SE-Bild konnten mehrere Tausend Partikel ausgewertet werden, was eine verlässliche Analyse der Partikelgrößen erlaubt. Hierdurch wurde es möglich, Nanopartikel sicher zu lokalisieren, identifizieren und ihre morphologischen Eigenschaften zu berechnen. Weiterhin wurden generative neuronale Netze eingesetzt, um eine Methode zu entwickeln, die künstliche Daten (SE-Bilder) erzeugt. Künstliche SE-Bilder können dann für das Training von neuronalen Netzen eingesetzt werden, um die vom Menschen erstellten Datensätze zu ersetzen. Künstliche Datensätze sind fehlerfrei und somit (theoretisch) menschlichen Datensätzen überlegen. Dies führt jedoch nicht zu den erwünschten Ergebnissen.

Zuletzt wurden die entwickelten Methoden verknüpft und typische REM-Bilder verschiedener Nanopartikel ausgewertet.

2 Theoretische Grundlagen des maschinellen Lernens

2.1 Tiefes maschinelles Lernen

Das tiefe maschinelle Lernen, kurz tiefes Lernen (*Deep Learning*), fasst eine Vielzahl von Algorithmen zusammen, welche in der Lage sind, deduktiv Wissen aus Beispielen abzuleiten. Diese Algorithmen erhalten während der Lernphase wiederholt Beispiele aus einem Datensatz und lernen Zusammenhänge in den Daten zu korrelieren. Dies erfolgt über die schrittweise Veränderung von internen, variablen Parametern (Gewichte) an die gezeigten Daten. Ziel des Trainings ist es, allgemeine Trends in den Daten zu entdecken und diese miteinander zu korrelieren, um „Wissen“ zu generieren. So kann schrittweise eine ideale Anpassung des Systems an die Daten gefunden werden. Die Anpassung des Systems wird regelmäßig über Testdaten überprüft. Sie ist dabei spezifisch für einen Typ von Daten, welche einen gemeinsamen Kontext besitzen, wie beispielsweise Bilder von Nanopartikeln. Ist das System ausreichend an die Trainingsdaten angepasst, wird das Training beendet. Die Parameter des Systems werden „eingefroren“. In der Anwendung auf neue Daten werden sich diese nun nicht weiter verändern.

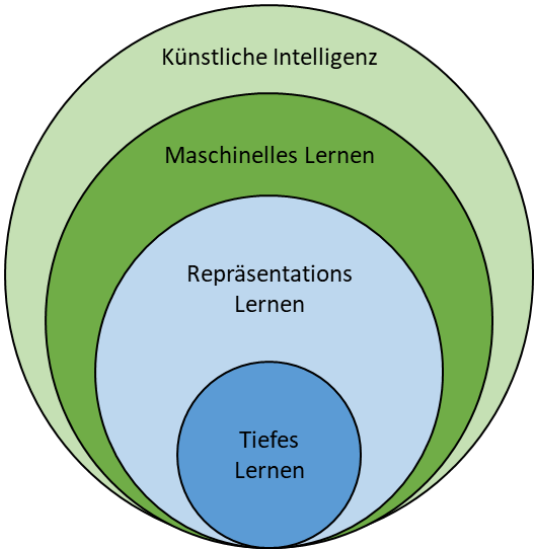


***Abbildung 1**: Zusammenhang der Teilgebiete der künstlichen Intelligenz. Das tiefe maschinelle Lernen ist die jüngste Entwicklung im Bereich der künstlichen Intelligenz. Während das einfache maschinelle Lernen auf schwache Zusammenhänge wie lineare Beziehungen zwischen Daten setzt, findet im tiefen Lernen eine Reihe hierarchisch aufbauender Abstrahierungen statt, um aussagekräftige Eigenschaften der Daten zu entdecken.* [25]

Für das Training eines Modells in Form eines KI-Systems sind mehrere Bestandteile erforderlich. Das System, welches für das Training genutzt wird, besteht aus drei Komponenten:

(i) Das eigentliche Modell $F(x, \theta)$ ist eine nicht lineare Funktion und der Teil des Systems, der für die Speicherung der Gewichte (θ) und die Verarbeitung von Daten (x) zuständig ist. Das Modell berechnet ein Ergebnis, welches auch Schätzung oder Vorhersage $(\hat{y})$ (*Prediction*) genannt wird. Das Modell ist eine Schätzfunktion. Um zu lernen und die Eigenschaften der Daten generalisieren zu können, nutzt das Modell Trainingsdaten.

(ii) Die bewertende Funktion $J\big(\hat{y}, F(x, \theta)\big)$ misst wie gut sich das Modell an die Testdaten angepasst hat. Die bewertende Funktion wird auch Verlust- oder Kostenfunktion (*Cost-, Loss-Function*) genannt. Das Ergebnis sind der Verlust bzw. die Kosten des Modells. Dieser Wert verhält sich umgekehrt proportional zum Bestimmtheitsmaß (R^2) in der Statistik. Ein perfektes Modell würde einen Verlust / Kosten von null zeigen.

(iii) Der Optimierungsalgorithmus passt, basierend auf der Abweichung des Modells vom wahren Ergebnis, die Gewichte des Modells weiter an die Trainingsdaten an.

Beim tiefen Lernen wird versucht, mittels eines Modells die Funktionsweise des Gehirns nachzubilden. Man folgt dabei der Annahme, dass ein Eingangssignal in mehreren Stufen von einem Ursprung (z. B. den Augen) bis zu einer Entscheidung verarbeitet wird. Es wird angenommen, dass die humane Verarbeitung von gesehenen Bildern stufenartig stattfindet und das Bilder lediglich abstrahiert, das heißt auf ihre inhaltliche Bedeutung reduziert, zu einer Entscheidung führen können. [26] Der Computer verarbeitet Eingangsdaten ebenfalls schichtartig und abstrahiert diese mittels des Modells. Das Modell ist ein Konstrukt aus einfachen mathematischen Funktionen, welche hierarchisch vernetzt sind. Zum Beispiel werden aus den Pixeln eines Bildes schrittweise abstrakte Zusammenhänge identifiziert, welche in bestimmter Orientierung, Zusammensetzung und Häufigkeit typisch für bestimmte Objekte sind (**Abbildung 2**). [27]

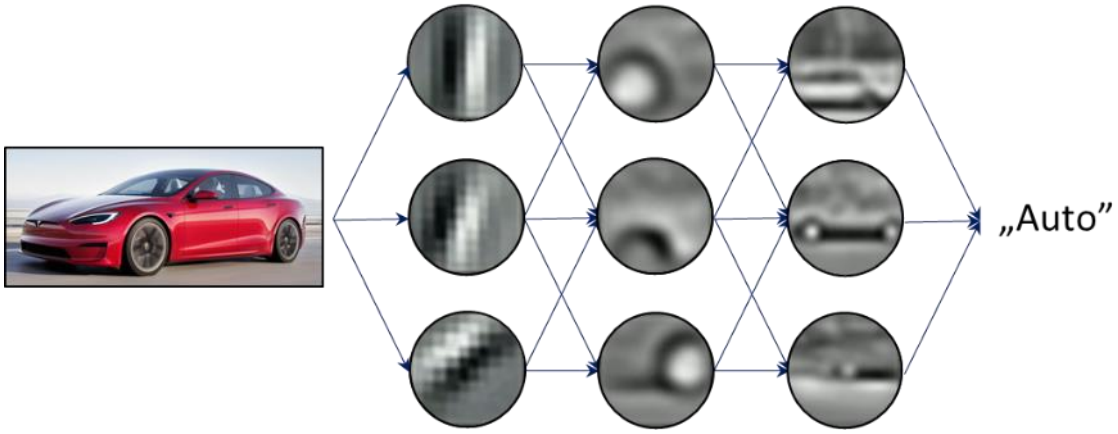


***Abbildung 2**: Die Abbildung zeigt schematisch die hierarchische Abstraktion des Bildinhaltes. Das „rohe“ Bild durchläuft eine Reihe von Schichten, die den Inhalt des Bildes schrittweise herausarbeiten. Es werden zunächst Kanten herausgearbeitet, bevor diese zu einfachen Elementen wie Ecken und weiter zu Umrissen verknüpft werden. Daraus ergibt sich die Semantik des Bildes, die das Modell nutzt, um eine Schätzung, was sich in dem Bild befindet, zu berechnen. Abbildung der Abstraktionen übernommen aus Lee et al. [28] Abbildung des Autos übernommen von Tesla Inc. [29]*

Als Grundlage des Modells fungieren künstliche Neuronen. Diese stellen die einfachste Recheneinheit im tiefen Lernen dar. Die Neuronen eines Modells beinhalten die variablen Parameter, welche während des Trainings angepasst werden. Ein solches künstliches Neuron wird Perzeptron genannt und ist in der Lage, ein Signal aufzunehmen (Input), zu verarbeiten und weiterzugeben (Output). [30, 31] Das Perzeptron ist eine mathematische Funktion, die ähnlich zur Funktionsweise echter Neuronen verschiedene Eingangsdaten miteinander korreliert. [32] Nach der Korrelation diverser Eingangsdaten gibt das künstliche Neuron ein Signal aus. Während es sich im Falle des biologischen Neurons um ein elektrisches Signal handelt, wird von einem künstlichen Neuron ein numerischer Wert ausgegeben. Das Perzeptron gibt das Signal in einer Intensität aus, die von der Gewichtung der Eingangsdaten abhängig ist. Die Gewichtung des Input ist das zentrale Element in der Funktionsweise des künstlichen Neurons. Die Werte, mit welchen künstliche Neuronen ihre Eingangsdaten gewichten, werden während des Trainings verändert. Perzeptronen lernen Eingangsdaten miteinander zu korrelieren. [32]

Die Verknüpfung von mehreren Perzeptronen ist die Grundstruktur eines neuronalen Netzes. Das Ausgangssignal des ersten Perzeptrons ist der Input des darauffolgenden Perzeptrons. Perzeptronen können auch parallel Eingangsdaten erhalten. Basierend auf der Hebbschen

Regel sollen Perzeptronen ein Eingangssignal verarbeiten und an folgende Perzeptronen weiterleiten. [33] Dabei werden Verbindungen von Perzeptronen, die in der Lernphase häufig gemeinsam Signale weiterleiten, verstärkt. Zusätzlich zur Weiterleitung starker Signale (positive Werte), werden schwache Signale (negative Werte) unterdrückt. Daher verfügt jedes Perzeptron über eine interne Schwelle, welche als Aktivierungsfunktion ($f_{Aktivierung}$) bezeichnet wird. **Formel 1** verdeutlicht den Aufbau einen Perzeptrons:

$$\hat{y}_i = f_{Aktivierung}\left(\sum_{i=0}^{n}(w_i * x_i) + b\right) \qquad 1$$

Die Parameter (θ), d. h. die Gewichte ($w_i \, \epsilon \, \theta$) sowie der Bias ($b \, \epsilon \, \theta$), sind während des Trainings eines Perzeptrons variabel. Die Verknüpfung von Perzeptronen zu einem neuronalen Netz erfolgt dabei in Schichten aus parallelen Einheiten, wobei jedes Perzeptron der ersten Schicht mit jedem Perzeptron der folgenden Schicht verbunden ist. Perzeptronen der ersten Schicht sind jedoch nicht mit weiteren Schichten verbunden. Daraus ergibt sich die Struktur eines gerichteten Netzes bzw. Graphen (*Feed Forward Net*) mit festem Start- und Endpunkt. [25] Schichten, welche zwischen Start- und Endpunkt liegen, werden verdeckte Einheiten (*Hidden Units*) genannt. Das Training dieser Netzwerke erfolgt über das Gradientenabstiegsverfahren (*Stochastic Gradient Descent, SGD*), [34] Die Anzahl der Inputs in ein Perzeptron, also der Verbindungen zu vorherigen Perzeptronen, ist beliebig wählbar. Je mehr Verbindungen zu vorherigen Neuronen bestehen, desto mehr variable Parameter enthält ein Neuron.

Die Art, auf welche Schichten aus Perzeptronen und weitere Bestandteile des Modells zusammengesetzt werden, wird als Architektur (*Architecture*) bezeichnet. Die Architektur ist entscheidend dafür, welche Daten das Modell verarbeiten kann und in welcher Form die Ausgangsdaten generiert werden. Verschiedene Architekturen werden für verschiedene Aufgaben wie die Lokalisierung oder Klassifizierung von Objekten entworfen. Dementsprechend unterscheiden sich die Ausgangsdaten von Architektur zu Architektur. Für die Verarbeitung von Bildern werden faltende neuronale Netze (*Convolutional Neural Networks, CNNs*) genutzt. Dieser Typ von neuronalem Netz nutzt Neuronen mit schwacher Konnektivität (*sparse connectivity*) zueinander. [25] Während in neuronalen Netzen die Verbindungen der Neuronen zueinander fixiert sind, können in faltenden neuronalen Netzen die Neuronen mit allen Pixeln des Bildes wechselwirken. Dadurch verändert sich auch die

Struktur der Neuronen. Sie werden 3-dimensional. Die zugrunde liegende mathematische Operation dieser Neuronen (auch Filter genannt) ist die diskrete Faltung. [35] Filter lernen Bildeigenschaften selektiv zu erkennen. [24] Bildeigenschaften sind unter anderem Pixelwerte, Pixelverteilung, Positionen von Pixelclustern und Gradienten im Bild wie Farbverläufe und Kanten in Form von starken Änderungen lokaler Pixelwerte. [36-38] Filter sind keiner festen Position von Pixeln zugeordnet und dadurch in der Lage, Eigenschaften eines Bildes unabhängig von ihrer Position zu erkennen. Folglich ist es möglich, rotierte, verzerrte und verschobene Objekte in Bildern zu identifizieren. [25]

Im Folgenden wird gezeigt, wie ein beliebiges neuronales Netz in der Lage ist, aus Daten zu lernen.

2.2 Training mittels Gradientenabstiegsverfahren

Werden neuronale Netze aus einer Vielzahl von Schichten aufgebaut, werden diese als tiefe neuronale Netze bezeichnet. Ab welcher Schichtanzahl ein neuronales Netz als „tief“ gilt, ist dabei nicht klar definiert. [25] Unabhängig von der Tiefe erfolgt das Training neuronaler Netze nach demselben Prinzip: Das Training neuronaler Netze beschreibt die iterative Anpassung der Modellparameter (θ) nach dem Zeigen der in einem Datensatz vorhandenen, Trainingsbeispiele. Trainingsbeispiele ermöglichen es dem Modell aus einer Reihe von Bildern, Texten oder Kundendaten zu lernen. [39] Das Training endet, sobald das Modell auf unbekannten Daten die gewünschte Leistung erbringt oder das Modell nicht mehr in der Lage ist, weiter zu lernen.

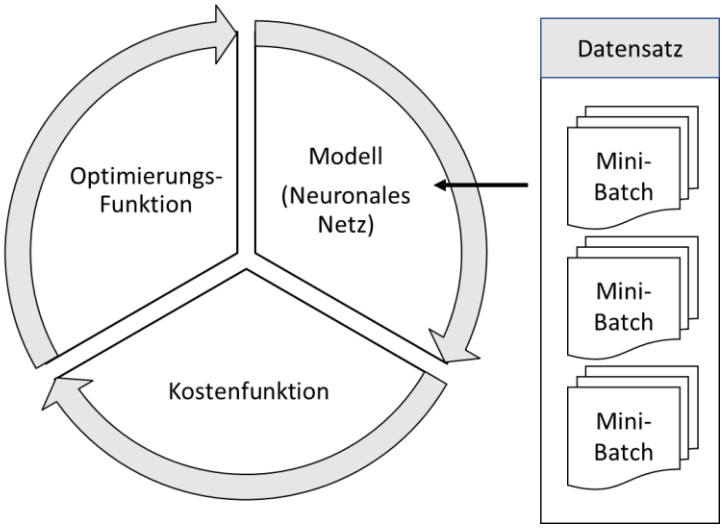


***Abbildung 3**: Das Gradientenabstiegsverfahren ist ein repetitiver Prozess, welcher aus dem wiederholten Zeigen aller Bilder eines Datensatzes in Form eines* Mini-Batch besteht. *Nach der Berechnung einer Wahrscheinlichkeitsverteilung durch das Modell werden die Kosten des* Mini-Batch *berechnet. Der Optimierungsalgorithmus passt die Gewichte des Modells basierend auf den Kosten des* Mini-Batch an. *Danach wiederholt sich der Kreislauf bis zur Konvergenz des Systems.*

Das Modell $F(x_i, \theta)$ ist eine nicht lineare Funktion, die sich aus Schichten künstlicher Neuronen sowie weiterer mathematischer Operationen, zusammensetzt. Das Modell nutzt gelabelte, das heißt von einem Menschen gekennzeichnete Trainingsbeispiele (x_i, y_i), um beispielsweise eine Bildbezeichnung mit seinem Inhalt semantisch zu verknüpfen. Bilder sind Matrizen mit diskreten Werten, welche durch ihre Farbtiefe gekennzeichnet sind. So besitzt

ein Graustufenbild (8 Bit) 256 verschiedene Graustufen inklusive schwarzer und weißer Pixel. Die Bildlabel wiederum werden vom Menschen als das benannt, was in einem Bild zu sehen ist. Ist auf einem Bild ein kubisches Nanopartikel zu sehen, so würde das Bild als „Würfel“ gekennzeichnet werden. Über eine Codierung kann dieses Label in einen binären Vektor übertragen werden (*One Hot Kodierung*). Die vektorielle Darstellung eines Labels ermöglicht es ein Bild und ein Label in einen mathematischen Zusammenhang zu stellen: $\hat{y}_i = F(x_i, \theta)$. Das neuronale Netz erhält ein Bild als Eingangssignal und berechnet für jedes mögliche Label eine Wahrscheinlichkeit, dass das Bild mit diesem Label versehen werden sollte (**Abbildung 4**). Die Summe der Wahrscheinlichkeiten aller Klassen beträgt 1 (100 %).

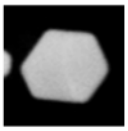
Bild	Menschliches Label	Kodierung	Model-Output
	„Stab”	$\begin{pmatrix}1\\0\\0\end{pmatrix}$	$\begin{pmatrix}1\\0\\0\end{pmatrix}$
	„Sechseck”	$\begin{pmatrix}0\\1\\0\end{pmatrix}$	$\begin{pmatrix}0\\0{,}14\\0{,}86\end{pmatrix}$
	„Kreis”	$\begin{pmatrix}0\\0\\1\end{pmatrix}$	$\begin{pmatrix}0\\0{,}10\\0{,}90\end{pmatrix}$

Abbildung 4: *Die Abbildung zeigt, wie elektronenmikroskopische Aufnahmen von verschiedenen Nanopartikeln von einem Menschen mit Labeln benannt wurden. Um Berechnungen mittels eines KI-Modells durchführen zu können, werden die Label in vektorielle Form gebracht (*One Hot Kodierung*). Das Modell berechnet eine Wahrscheinlichkeitsverteilung, dass das Bild zu einer der trainierten Klassen gehört. Der Output eines Modells, welches darauf trainiert wurde, die drei gezeigten Formen zu erkennen, könnte so lauten wie hier gezeigt. „Stab“ und „Kreis“ wurden korrekt erkannt, während „Sechseck“ falsch erkannt wurde. Die reale Größe der Partikel ist unbedeutend.*

Über die Kodierung der Bildklassen in eine vektorielle Form kann die Anpassung des neuronalen Netzes gemessen werden. Je besser das neuronale Netz die wahre Klasse berechnen kann, das heißt, je höher die Wahrscheinlichkeit für die korrekte Klasse ist, desto geringer ist der Fehler. Der Fehler wird mittels einer Verlust- bzw. Kostenfunktion

$J(y_i,\ F(x_i,\theta))$ berechnet. Das Ziel (*Objective*) des Trainings ist es über Anpassung der Parameter, innerhalb der Neuronen, die Kostenfunktion zu minimieren. Das Verfahren, um dies zu erreichen, wird Gradientenabstiegsverfahren genannt. [34] Je geringer die Kosten des Modells sind, desto höher ist der Grad der Anpassung an den Datensatz. Die Anpassung des Modells ist dabei spezifisch an den, im Training verwendeten, Datensatz. Das Modell lernt von einem Datensatz auf die Eigenschaften der Daten zu generalisieren. Sofern 2 unterschiedliche Datensätze, unterschiedliche Eigenschafts-Räume besitzen, kann ein Modell, das auf Datensatz A trainiert wurde, nicht auf Datensatz B angewendet werden.

Für die Berechnung der Kosten eines Modells ist die Wahrscheinlichkeitsverteilung für alle Bilder des Datensatzes erforderlich. Da aber der benötigte Speicherbedarf bei großen Datensätzen technisch nicht zu realisieren ist, wird der Fehler einer Teilmenge von Bildern des Datensatzes berechnet (Mini-Batch). Ein Mini-Batch hat eine Größe n Bildern. n liegt typischerweise zwischen 1 und 256 Bildern. Komplexere Netzwerke können auch eine Größe von ca. 2000 Trainingsbeispielen pro Mini-Batch besitzen. [40] Nachdem das Modell die Wahrscheinlichkeitsverteilung für alle Bilder des Mini-Batches berechnet hat, wird die Kostenfunktion angewendet. Mini-Batches werden im Training nacheinander abgearbeitet.

Es existieren eine Vielzahl von Kostenfunktionen. [41] Die hier verwendeten Netzwerke nutzen die Kreuzentropie, welche auf der *Log-Likelihood* Funktion basiert. [42]

$$J(y_i, F(x_i,\theta)) = \frac{1}{n}\sum_{i}^{N} -y_i * \log(\hat{y}_i) - (1-y_i) * \log(\hat{y}_i) \qquad 2$$

Die *Log-Likelihood* Funktion entstammt der Informationstheorie. [43] Sie misst den mittleren Abstand zwischen dem wahren Ergebnis (y_i) und der Vorhersage des Modells ($\hat{y}_i$), für einen *Mini-Batch*, wobei n für die Größe des *Mini-Batch* steht (Anzahl der gezeigten Beispiele). Die Kreuzentropie konnte bereits mehrfach zeigen, dass sie in verschiedenen neuronalen Netzen als Kostenfunktion genutzt werden kann. [44, 45] Bei der Annäherung des vom Modell berechneten Wertes $\hat{y}_i$ an den wahren Wert y_i, sinkt der Fehler logarithmisch. Viele Netzwerke können die Kreuzentropie als Kostenfunktionen verwenden. [41, 46] Netzwerke mit speziellen Aufgaben wie der Generierung von Bildern haben spezielle Kostenfunktionen. [47]

Die Anzahl der Bilder im Datensatz, wie auch die Anzahl von Bildern pro Klasse und die Größe des *Mini-Batch haben* einen Einfluss auf das Lernverhalten des neuronalen Netzes. Werden in

einem *Mini-Batch* lediglich Bilder einer Klasse gezeigt, kann das Netzwerk nur Eigenschaften von dieser Klasse erlernen. Ein *Mini-Batch* muss daher immer eine ausgewogene Menge von Bildern aller im Datensatz enthaltenen Klassen besitzen. Sollte dies nicht möglich sein, da beispielsweise eine ungleiche Menge von Bildern pro Klasse im Datensatz vorhanden ist, muss die Kostenfunktion modifiziert werden. Eine ungleiche Klassenverteilung ist ein reales Problem in vielen Datensätzen aufgrund mangelnder Daten zu bestimmten Klassen. So enthält der für diese Arbeit erstellte Datensatz der SE-Bilder relativ wenige Würfel. Im Training würde dies bedeuten, dass das Modell relativ wenige kubische Objekte „sieht“. Die Eigenschaften für kubische Objekte wären somit unterrepräsentiert und das Modell kann sich nicht an diese anpassen. Die Lösung dieses Problems ist die Nutzung einer Kostengewichtung (*Dataset-Weighting, Loss-Weighting*). [24] Eine Kostengewichtung greift direkt in die Kostenfunktion ein und gewichtet dabei den Einfluss jedes Bildes in Abhängigkeit zur Anzahl der Bilder seiner Klasse im Datensatz. Je größer der Anteil von Bildern einer Klasse im Datensatz, desto geringer ist der Wert eines einzelnen Bildes. **Formel 3** wird dabei mit einem Gewichtungsterm (λ_j) für jede Klasse (j) modifiziert. Der Gewichtungsterm wird mittels **Formel 4** berechnet.

$$J\big(y_i, F(x_i, \theta)\big) = \frac{1}{n}\sum_{i}^{N} -y_i * \log(\hat{y}_i) * \lambda - (1 - y_i) * \log(\hat{y}_i) * (1 - \lambda) \qquad 3$$

$$\lambda_j = \left(\frac{N}{M}\right)^{-1/2} \qquad 4$$

N beschreibt dabei die Anzahl der Datenpunkte (Bilder) pro Klasse und M die Anzahl aller Datenpunkte im Datensatz [44] Dies simuliert dem Netzwerk durch Anpassen der Kostenfunktion einen ausgeglichenen Datensatz und resultiert in einem effizienteren Training.

Bis zu diesem Punkt hat das System nichts gelernt. Es wurde der mittlere Fehler des Modells berechnet, nachdem das Modell eine Wahrscheinlichkeitsverteilung für jedes in einem *Mini-Batch* vorhandenen Bild berechnet hat. Um die Parameter des Modells anzupassen, muss der Einfluss jedes Parameters auf das Ergebnis des Modells bestimmt werden. Da es sich bei der Kostenfunktion wie auch dem Modell um eine differenzierbare Funktion handelt, kann das Minimum der Kostenfunktion über ihre Ableitung gefunden werden. Der Einfluss einzelner Parameter wird über die partielle Ableitung nach den im Modell verwendeten Parametern berechnet. Die partielle Ableitung nach den Parametern eines Systems wird auch Gradient

(∇_Θ) genannt und beschreibt den Einfluss der Parameter auf die Leistung des Modells. Der Gradient des Systems entspricht dabei **Formel 5**.

$$\nabla_\theta J_\theta = \frac{\partial F}{\partial \theta_i} J\big(y_i, F(x_i, \theta)\big) \quad 5$$

Nachdem für jeden im Modell vorhandenen, Parameter der Gradient (∇_Θ), über seine partielle Ableitung bestimmt wurde, werden diese mit einem Optimierungsalgorithmus angepasst. **Formel 6** zeigt dabei die einfachste Form der Parameteranpassung:

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_\theta J_\theta \quad 6$$

Die Parameter θ_t leiten sich aus den Parametern des vorherigen Schrittes θ_{t-1} minus des Gradienten $\nabla_\theta J_\theta$ ab. Der Term η ist die Lernrate (*Learning Rate*) und bestimmt, wie stark die Parameter des Systems nach dem Gradienten angepasst werden. Die Anpassung der Gewichte erfolgt nach jedem Mini-Batch. Ein einziger Mini-Batch enthält jedoch nicht alle Eigenschaften eines Datensatzes. Zudem sind Bilder oftmals verrauscht oder der Hintergrund beeinflusst die Wahrscheinlichkeitsverteilung des Modells. Das Modell lernt diese irrelevanten Faktoren zu ignorieren. Innerhalb eines Datensatzes kommen viele Eigenschaften vor. Mache werden dabei in der einen Parameterkonfiguration besser identifiziert als in der anderen. Durch die Anpassung der Gewichte nach jedem *Mini-Batch* kann eine Konfiguration gefunden werden, mit welcher alle Eigenschaften eines Datensatzes identifizierbar sind. Parameter wie die Lernrate oder Klassengewichtungen, welche die Leistung des Modells zwar beeinflussen, aber nicht aktiver Teil des Modells sind, werden Hyperparameter genannt (*Hyperparameters*). **Formel 6** wird Optimierungsfunktion (*Optimizer*) genannt, da diese Funktion die Parameteroptimierung übernimmt. Die hier verwendeten Netzwerke nutzen alle den ADAM-Algorithmus zur Anpassung der Gewichte. [46] Die einfache Parameteranpassung ist anfällig gegenüber Sattelpunkten der partiellen Ableitungen. Der Gradient an einem Sattelpunkt ist null. Das Training würde also an Sattelpunkten stoppen. Der ADAM-Algorithmus ist in der Lage, dies zu verhindern.

Nachdem die genannten Berechnungen von Modell, Kostenfunktion und Optimierungsfunktion für einen *Mini-Batch* ausgeführt wurden, wird so lange mit weiteren Mini-Batches trainiert, bis alle Bilder des Datensatzes genutzt wurden, um die Kosten zu berechnen. Eine Trainingsepoche endet daraufhin. Nach einer Trainingsepoche wird das Modell validiert. Bilder, die für die Validierung genutzt werden, sind zufällig ausgewählte

Bilder jeder Klasse, die vor dem Training von den übrigen Bildern getrennt wurden. Die Validierungsbilder werden nicht für das Training genutzt und dienen ausschließlich zur Überprüfung des Netzwerkes. Da das Netzwerk die Eigenschaften dieser neuen Bilder noch nie „gesehen“ hat, kann so überprüft werden, wie gut das Netzwerk die Eigenschaften des Datensatzes generalisiert. Neben den Kosten werden in diesem Schritt auch andere Metriken verwendet, um eine Aussage über die Nutzbarkeit des Modells treffen zu können. Typischerweise wird hierfür die Korrektklassifizierungsrate verwendet, die den relativen Anteil der richtig erkannten Validierungsbilder angibt.

Nach der Validierung ist das Training nicht abgeschlossen. Die Bilder des Datensatzes werden einem Kartenstapel gleich gemischt und in neue Mini-Batches aufgeteilt. Es beginnt eine neue Trainingsepoche. Die Anpassung der Parameter wird für mehrere Epochen fortgeführt, bis sich die Kosten der Validierungsbilder nicht mehr ändern. Das Training wird an diesem Punkt beendet. Die Parameter des Modells werden eingefroren und nicht mehr verändert.

Der gesamte Algorithmus umfasst 4 Schritte, bis sich die Kosten des Modells nicht mehr verringern:

1. Dem System werden die Wertepaare x, y des Datensatzes D zur Verfügung gestellt.
2. Das Modell errechnet eine Schätzung $F(x_i, \theta) = \hat{y}_i$ für jedes Trainingsbeispiel x_i eines Mini-Batches.
3. Die Kostenfunktion wird nach allen Parametern θ des Systems partiell abgeleitet, um den Gradienten ∇_θ zu bestimmen.
4. Die Optimierungsfunktion wendet den negativen Gradienten nach **Formel 6** an, um die Parameter zu optimieren.

   Die Schritte 2 bis 4 werden wiederholt, bis keine Verbesserung der Leistung mehr stattfindet.

Die Parameteroptimierung nach dem Gradientenabstiegsverfahren ist die einzige effektive Methode zum Trainieren eines neuronalen Netzes, unabhängig von seiner Art. [25] Mit den gezeigten Bausteinen: Modell, Kostenfunktion und Optimierungsfunktion wird jedes neuronale Netz trainiert. Dabei ist jeder Baustein individuell und gegen eine andere Funktion gleicher Art austauschbar. Die hier verwendete Kreuzentropie als Kostenfunktion und ADAM als Optimierungsalgorithmus sind jeweils nur ein Vertreter ihrer Art. Je nach Typ des Modells

müssen Kostenfunktion und Optimierungsfunktion angepasst werden, um zufriedenstellende Ergebnisse zu erhalten [48]

2.3 Faltende neuronale Netze

Seit der Erfolge von AlexNet [24] im Jahre 2011 sowie leistungsstärkeren Grafikkarten dominieren faltende neuronale Netze (CNN) in Bereichen der Bildverarbeitung, Mustererkennung, Spracherkennung, Übersetzung, Spieltheorie und chemischer wie biologischer Modellierungen. [39]

Moderne faltende neuronale Netze (CNNs) bestehen aus einer Reihe von hierarchisch ausgeführten Operationen. Diese werden Schichten (*Layers*) genannt. Eine Schicht fasst mehrere mathematische Operationen in einer gemeinsamen parallelen Einheit zusammen, sodass allgemeine Instruktionen, wie z. B. die Parameterinitialisierung [49, 50] für eine gesamte Schicht gelten.

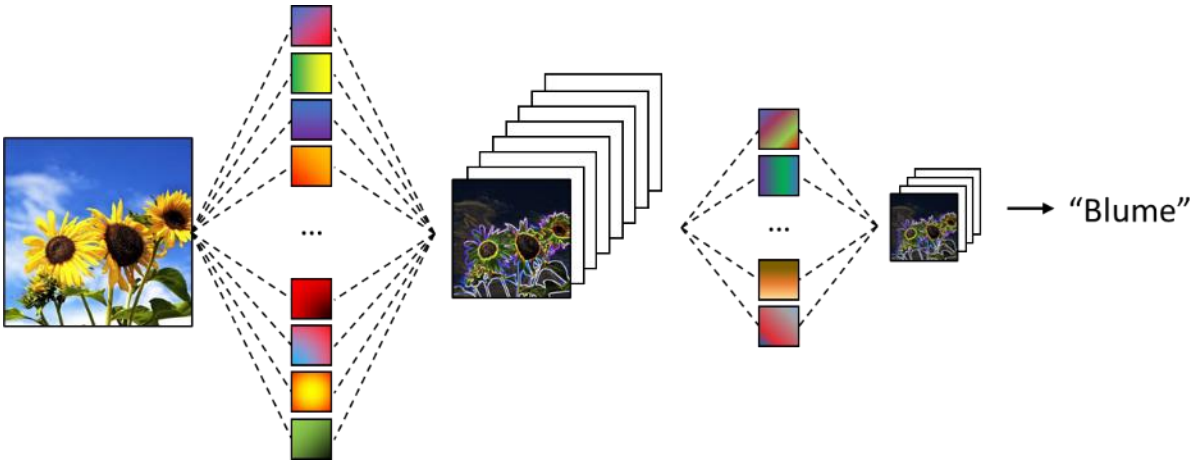


***Abbildung 5**: Hierarchischer Aufbau eines faltenden neuronalen Netzes. Das Bild wird von mehreren individuellen Filtern auf seine Eigenschaften untersucht. Die Filter dieser ersten Schicht sind einfache Kanten- & Farbdetektoren. Die individuellen Eigenschafts-Karten heben jeweils eine Eigenschaft hervor, auf die sich der Filter angepasst hat. Diese Eigenschafts-Karten werden konkateniert und an die nächste Schicht weitergegeben. Die nächste Schicht untersucht den gesamten Eigenschafts-Tensor auf die Beziehungen der einzelnen Eigenschaften zueinander. Ihre Filter sind daher deutlich komplexer als die der ersten Schicht. Wiederum gibt jeder Filter der zweiten Schicht eine einzelne Eigenschafts-Karte aus, welche konkateniert, den Eigenschafts-Tensor dieser Schicht bilden. Dieser Prozess wird Eigenschaftsextraktion genannt. Durch die Faltungen schrumpft zudem die Größe des Bildes.*

Jede Schicht des Netzwerkes besteht aus einer Reihe von Filtern. Die Filter einer Schicht können dabei als Eigenschafts-Extraktoren verstanden werden. Jeder Filter untersucht das

Bild auf eine Eigenschaft (*Feature*) und berechnet aus dem Filter und dem Bild eine Eigenschafts-Karte (*Feature Map*). Eigenschafts-Karten zeigen, wo eine Eigenschaft im Bild auftritt. [26] Eigenschafts-Karten sind Matrizen, welche Bereiche kennzeichnen, die den Filter ansprechen und daraus resultierend dem Modell Informationen über den Bildinhalt liefern. Durch die Anwendung vieler Filter in einer Schicht und die gestapelte Ausgabe der von den Filtern berechneten Eigenschafts-Karten ergibt sich ein Eigenschafts-Tensor (*Feature-Tensor*). Jede Schicht gibt dabei einen Eigenschafts-Tensor aus, welcher als Input der folgenden Schicht fungiert. Die darauffolgende Schicht verknüpft die im Tensor dargestellten Eigenschaften und gibt erneut einen Eigenschafts-Tensor aus. Durch die lineare Verkettung dieser Schichten entsteht ein hierarchischer Aufbau von Filtern, die sukzessiv größere Teile des Eingangsbildes semantisch in Korrelation bringen (**Abbildung 5**). [51]

Der Begriff der Schicht ist dabei nicht nur auf trainierbare Operationen beschränkt. Auch werden Operationen zur Dimensionsreduktion, Konkatenation und Normalisierung von *Eigenschafts-Tensoren* als Schichten bezeichnet. Jedoch ist der Begriff „Schicht“ historisch und zum Teil irreführend. Erste Netzwerke waren streng linear und nutzten ausschließlich Schichten verknüpfter Perzeptronen [30], während moderne faltende neuronale Netzwerke viele Operationen parallel ausführen. [52] Schichten können daher auch als Knoten eines gerichteten Graphen verstanden werden. Auch können durch das Zusammenfassen von Operationen in Modulen unterschiedliche Operationen in einem CNN genutzt werden, um die Eigenschaften (*Features*) eines Bildes auf unterschiedlichen Skalen zu untersuchen. Die Ausgangstensoren dieser parallelen Sub-Schichten werden im Anschluss zu einem einzelnen *Eigenschafts-Tensor* konkateniert. Ein typisches Beispiel sind die Inception-Module von GoogLeNet. [23]

Jede Schicht besteht dabei aus einem Input der eigentlichen Operation, welche auf den Input angewendet wird, sowie dem Output. Input und Output können dabei unterschiedliche Größen haben. Dies bedeutet, dass eine Schicht, die ein Bild als Input hat, nicht zwangsläufig ein Bild generiert. Vielmehr generieren die meisten Schichten eines CNNs einen Stapel von Bildern, wobei jedes Bild für sich eine repräsentative Aufnahme des Inputs ist (siehe **Abbildung 5**). Da ein neuronales Netz die Eigenschaften eines Bildes kodiert, nennt man die von Schichten generierten Vektoren und Matrizen Eigenschafts-Tensoren.

2.3.1 Schichten eines faltenden neuronalen Netzes

Vollvernetzte Schicht (Fully Connected Layer, FC-Layer)

Vollvernetzte Schichten sind die „klassischen“ Schichten eines neuronalen Netzwerks und bestehen aus einer Ebene von Perzeptron. Um den Eigenschafts-Tensor einer Convolutional-Schicht oder einen Bild-Tensor in eine vollvernetzte Schicht einfügen zu können, muss dieser zuvor in einen Vektor umgewandelt werden.

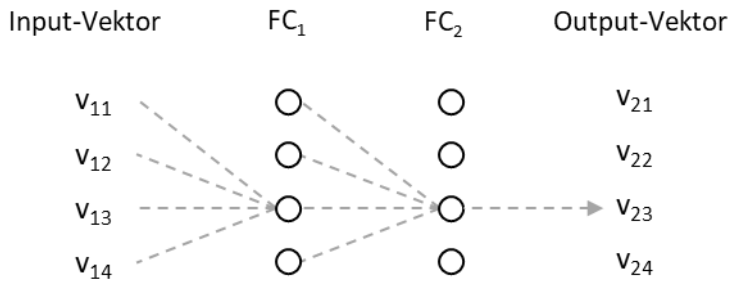


***Abbildung 6**: Schematische Darstellung von zwei miteinander verknüpften vollvernetzten Schichten. Jedes der 4 Neuronen der Schicht FC_1 ist mit jedem Wert des Inputs verbunden (siehe **Formel 1**). Folglich ist jedes Neuron der Schicht FC_2 mit jedem Neuron der Schicht FC_1 verbunden. Da nach der FC_2-Schicht keine weitere Operation folgt, werden alle Ausgabe-Werte der Schicht FC_2 als ein Vektor ausgegeben.*

Die Ausgabe einzelner Neuronen ist abhängig von allen Werten der vorangegangenen Schicht und den Gewichten (θ) des Neurons. Jedes Neuron verfügt zudem über eine Aktivierungsfunktion, um zu bestimmen, ob die gebündelten Eingangssignale propagieren. Die Aktivierungsfunktion für alle Neuronen einer Schicht ist gleich und entspricht in den meisten Klassifizierungs-Netzwerken der Gleichrichterfunktion (*Rectified Linear Unit, ReLU*). [24, 53] In Klassifizierungsnetzwerken bilden vollvernetzte Schichten die letzte Schicht des Netzwerkes. Diese geben dann die Wahrscheinlichkeit für jede trainierte Klasse aus. Die Aktivierungsfunktion in der letzten Schicht ist die Sigmoid-Funktion.

Faltungs-Schicht (Convolutional-Layer)

Moderne neuronale Netze bestehen zu großen Teilen, wenn nicht sogar vollständig, aus Faltungs-Schichten (*Convolutional-Layers*). Die mathematische Operation der Faltung (✱) einer Funktion f auf eine Funktion g ist dabei gegeben als:

$$f(x) \ast g(x) = \int_0^{\tau} f(\tau) * g(x-\tau)\, d\tau \qquad 7$$

Im Bereich des maschinellen Lernens wird mit Vektoren bzw. Matrizen, allgemein Tensoren gearbeitet. Hierdurch ergibt sich folgende Notation, in der das Signal x mittels des Filters w verfaltet wird. x und w sind Vektoren, für deren Elemente gilt $x_i, w_i \in \mathbb{R}$. Die Elemente eines Eingangsvektors x oder Ausgangsvektors y nennt man auch Merkmale, und den dementsprechenden Vektor „Eigenschafts-Vektor“ (*Feature-Vector*).

$$y = x \ast w \;\rightarrow y_i = \sum_{k=-\infty}^{\infty} x_{i-k} * w_i \qquad 8$$

Die Faltung findet theoretisch über den gesamten Bereich von $-\infty$ bis $+\infty$ statt. Da jedoch nur endliche Tensoren existieren, werden die Bereiche außerhalb des Vektors mit Nullen „aufgefüllt“. Diese Technik wird *Zero-Padding* genannt. Die Anzahl der Werte des *Paddings* entscheidet in der Faltung über die Größe des Ausgangsvektors.

Im Fall einer Matrix X (Bild), die wiederum mit einer Matrix W (Filter) verfaltet wird, ändern sich die verwendeten Indizes, da über 2-Dimensionen gerechnet wird:

$$Y = X \ast \mathrm{W} \rightarrow \mathrm{Y_{i,j}} = \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} X_{i-k_1, j-k_2} * W_{k_1, k_2} \qquad 9$$

Wird **Formel 9** auf einen Tensor mit beliebiger tiefe angewendet, so muss für die Faltung von X ein Filter W mit selbiger tiefe verwendet werden. Die Matrix X und der Filter W haben jeweils eine Größe, was der Anzahl ihrer Elemente in x, y-Richtung entspricht. Die Größe von X ist um ein Vielfaches größer als W. Da die Tiefe eines Filters der Tiefe des Input-Tensors entspricht, wird dies bei der Benennung eines Filters bzw. ganzer Schichten vernachlässigt. Auch für 3-dimensionale Tensoren wie Bilder wird der Rand des Bildes mit Nullen aufgefüllt, um die Größe des Ausgangstensors konstant zu halten. **Abbildung 7** demonstriert dies.

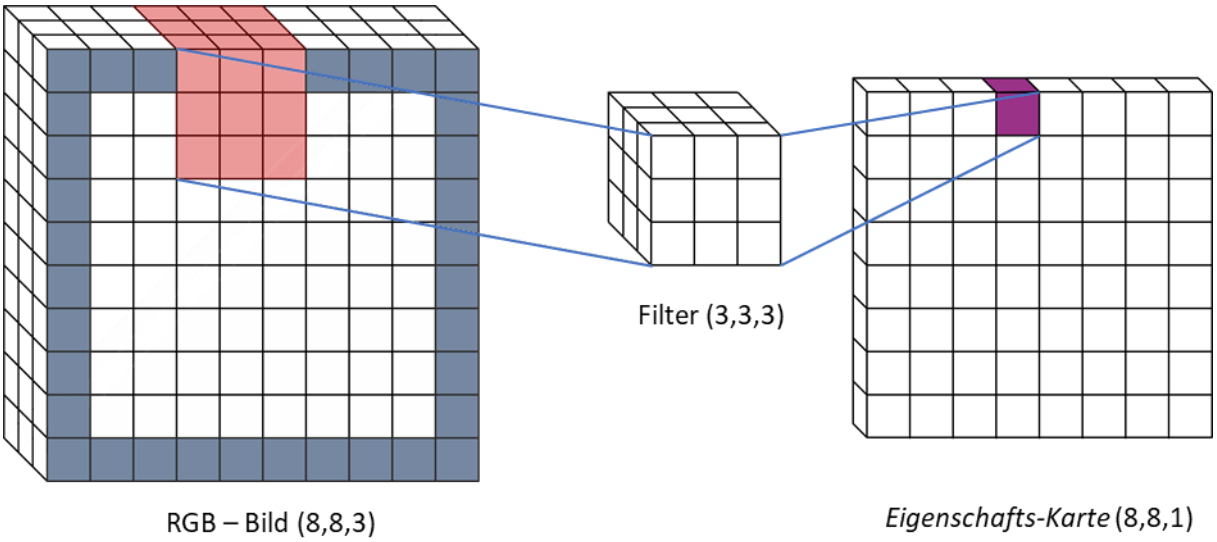


***Abbildung 7**: Schematische Darstellung der Faltung eines farbigen 8x8 Pixel großen Bildes mit den Farbkanälen Rot, Grün und Blau, welche als gestapelte Matrizen vorliegen (Eingangs-Tensor). Der Eingangs-Tensor besitzt die Form (8,8,3). Grau markierte Felder zeigen zum Bild hinzugefügte Pixel aus Nullwerten (*Zero-Padding*). Dies führt dazu, dass sich die Größe des Bildes während der Faltung nicht verändert. Der rote Bereich zeigt schematisch den Bereich des Bildes, welcher mittels des Filters verfaltet wird, um ein Pixel der Eigenschafts-Karte zu berechnen. Der Filter hat eine Größe von 3x3 Werten und entsprechend des Input-Tensors eine Tiefe von 3. Der generierte Output ist eine Eigenschafts-Karte welche die Größe (8,8,1) aufweist. Der Filter fährt das Bild in x, y-Richtung ab, um eine vollständige Eigenschafts-Karte zu erzeugen.*

Im Prozess der Faltung wird der Mittelpunkt des Filters, dem *Kernel*, auf jedes x, y-Koordinatenpaar des *Inputs* gelegt. Es findet eine Matrixmultiplikation der übereinanderliegenden Wertepaare des *Input-Tensors* sowie des Filters statt. Das Ergebnis wird auf der korrespondierenden Position der Eigenschafts-Karte hinzugefügt und der Filter um einen Pixel verschoben. Typischerweise werden mehrere Filter pro Schicht genutzt. Die generierten Eigenschafts-Karten aller Filter werden gestapelt, welcher dann als ein gemeinsamer Eigenschafts-Tensor dieser Schicht fungiert. Somit entspricht die Anzahl der Filter einer Schicht der Tiefe des erzeugten Eigenschafts-Tensors.

Der Vorteil von Faltungs-Schichten gegenüber Vollvernetzten-Schichten wird in der Vernetzung ersichtlich. Im Gegensatz zu einem Perzeptron, dessen Input an ein festes Pixel gebunden ist, sind Filter mobil und durchlaufen eine Verbindung zu jedem Pixel des Bildes. Da

die Filter einer Faltungs-Schicht deutlich kleiner sind als das zu verfaltende Bild, besteht eine spärliche Konnektivität (*Sparse Conectivity*) zwischen Filter und Input. [25] Der Einsatz von mehreren Filtern auf einen Input ermöglicht somit den gleichen Informationsgewinn bei deutlich geringerer Parameteranzahl. So würde ein Graustufen-Bild der Größe 32x32 px (32*32 = 1024 Einträge) in einer FC-Schicht 1,05 Millionen Parameter benötigen, um einen 1024 Einträge enthaltenden Eigenschafts-Tensor zu erzeugen. Dagegen benötigt eine Faltungs-Schicht mit 103 3x3 Filtern lediglich 1030 Parameter, um einen Eigenschafts-Tensor selbiger Größe zu generieren.

Konzentrations-Schicht (MaxPool-Layer)

Aufgrund der genutzten Aktivierungsfunktionen in neuronalen Netzen propagiert ein Signal, wenn es größer null ist. Die führt unweigerlich dazu, dass der Output von Faltung- oder Vollvernetzten-Schichten viele Nullwerte enthält. Um das Signal „aufzukonzentrieren“, werden Maximum Pooling-Schichten (*MaxPool*) genutzt. Diese agieren ähnlich wie Faltungs-Schichten, d. h. ein Filter rastert mit einer entsprechenden Größe und Schrittweite den Input-Tensor ab. Anders als Faltungs-Schichten hat der Filter einer Pooling-Schicht keine eigenen Werte, sondern überträgt das stärkste Signal des abgetasteten Bereichs auf den Ausgangstensor. Die Tiefe des Input-Tensors wird nicht verändert, während die x, y-Ausdehnung abnimmt. Auch nutzen Pooling-Schichten kein Padding, wenn Folgendes gilt:

$$n = \frac{X_{Größe}}{W_{Größe}} \qquad 10$$

n entspricht einer positiven, natürlichen Zahl. Andernfalls wird das Bild mit einem entsprechenden *Zero-Padding* versehen, sodass $n \in \mathbb{N}$ erfüllt ist.

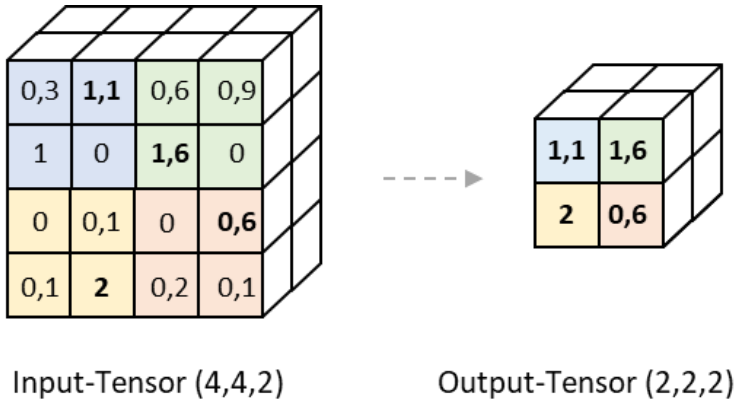


***Abbildung 8**: Anwendung eines 2x2s2 Pooling Filters auf einen Input Tensor der Größe (4,4,2). Die unterschiedlichen Farben zeigen jeweils, in welchem Bereich der Filter das maximale Signal propagiert. Der Output ist entsprechend kleiner als der Input.*

2.3.2 Module eines faltenden neuronalen Netzes

Wie bereits erwähnt, können Schichten innerhalb eines CNN zu Modulen bzw. Blöcken zusammengeschlossen werden. Module dienen der einfachen Darstellung repetitiver Schichten. Die hier trainierten Netzwerke beinhalten die im Folgenden beschriebenen Module. Die Hauptaufgaben der Module sind:

(i) Überspringen von Schichten mit anschließender Rückführung eines kopierten Tensors. [49]

(ii) Repetitive Faltung und Verkettung von Eigenschafts-Tensoren. [54]

Dies hilft dem Netzwerk, Bildeigenschaften auf verschiedenen Skalen zu untersuchen.

Identitäts-Module (Identity-Module)

Die Architektur Residual-Network (ResNet) führte Identitäts-Module (*Identity-, Residual-Module*) ein (**Abbildung 9**). [49] Innerhalb von Identitäts-Modulen wird der Eingangstensor zunächst kopiert und anschließend durch mehrere Faltungs-Schichten geführt. Im Anschluss findet eine elementweise Addition der verfalteten Kopie sowie des ursprünglichen Eingangstensors statt. Durch diese Technik ist das Netzwerk in der Lage, effektiver zu trainieren.

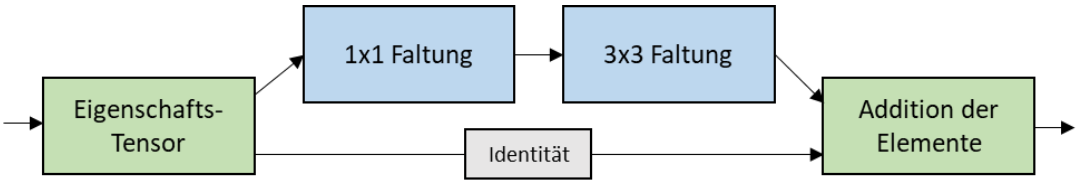


***Abbildung 9**: Beispiel eines Residual-Moduls wie es in* ResNet34 *zur Anwendung kommt.* [19]

Dichte Faltungsmodule (Dense Convolutional Layer)

Der Nutzen von mehreren Schichten bestehend aus 3x3 Filtern gegenüber einzelnen Schichten, bestehend aus 11x11 oder 7x7 Filtern, liegt an der geringeren Anzahl von Parametern bei gleichem rezeptivem Feld (*Receptive Field*). Das rezeptive Feld ist der Bildbereich des Eingangsbildes, der Einfluss auf ein Pixel einer Eigenschafts-Karte nimmt. Eigenschafts-Karten aus tieferen Schichten des neuronalen Netzes besitzen ein größeres rezeptives Feld als Eigenschafts-Karten aus flacheren Schichten. Erst durch die Zunahme des rezeptiven Feldes mit der der Tiefe des Netzwerkes ist es möglich, größere Bereiche des Bildes in einen semantischen Zusammenhang zu stellen. Ein Bild, welches mittels gestapelter Schichten kleiner Filter analysiert wird, erfährt die gleiche Signalabtastung wie ein Bild, das mit einer Convolutional-Schicht großer Filter abgetastet wird. [54]

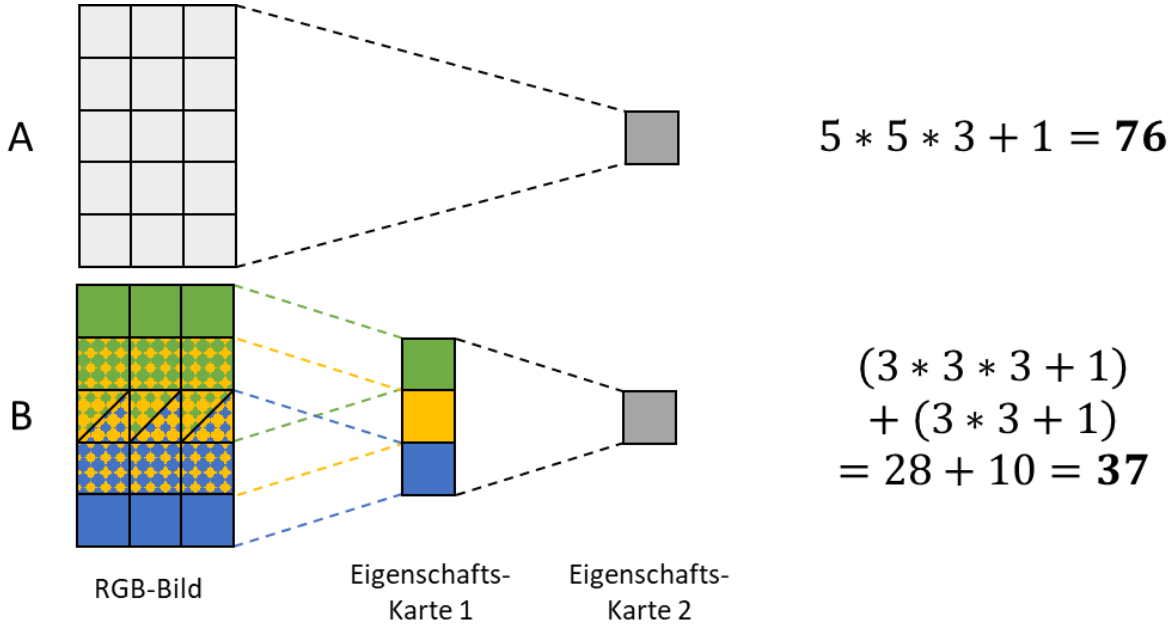


***Abbildung 10**: Die Abbildung zeigt den Unterschied der Anwendung von 2 aufeinander folgenden kleinen Filtern der Größe 3x3, gegenüber der Anwendung eines größeren Filters der*

*Größe 5x5. Der Input ist ein RGB-Eingangsbild mit einer Größe von 5x5 Pixeln. Das Bild besitzt 3 Farbkanäle (Rot, Grün und Blau). Der Bild-Tensor ist von der Seite entlang der y-Achse betrachtet. Das rezeptive Feld ist farblich markiert. So haben alle gelb markierten Pixel des RGB-Bildes einen Einfluss auf den gelben Wert der Eigenschafts-Karte 1. Ein Filter, welcher die Größe 5x5 besitzt, würde 76 Werte besitzen, um das Bild verfalten zu können. 75 Werte in Form von Gewichten für die RGB-Werte des Bildes sowie 1 Bias Term (**A**). Die Anwendung von 2 aufeinander folgenden 3x3 Filtern würde weniger Gewichte besitzen. Der erste Filter reduziert das Bild zur Eigenschafts-Karte 1. Dafür benötigt er 27 Gewichte für Gewichte für die RGB-Werte des Bildes sowie 1 Bias Term. Filter 2 faltet Eigenschafts-Karte 1 weiter zu Eigenschafts-Karte 2. Dafür benötigt er 9 Gewichte, für die Werte der Eigenschafts-Karte 1sowie 1 Bias Term. Insgesamt benötigt die Anwendung von 2 kleinen Filtern 37 Parameter. (**B**)*

Die Anwendung von mehreren kleinen, aufeinander aufbauenden, Filtern führt zur selben Verarbeitung des Bildes wie ein großer Filter (**Abbildung 10**). Daraus ergibt sich die Struktur des dichten Faltungsmoduls (*Dense Convolutional Layer, DC-Module).* [54] DC-Module generieren letztendlich einen Eigenschafts-Tensor, der die Eigenschaften großer und kleiner Bildbereiche semantisch vereint. Vor allem in der Segmentierung sind dichte Faltungsmodule hilfreich, um z. B. diffuse Ränder von Objekten zu untersuchen.

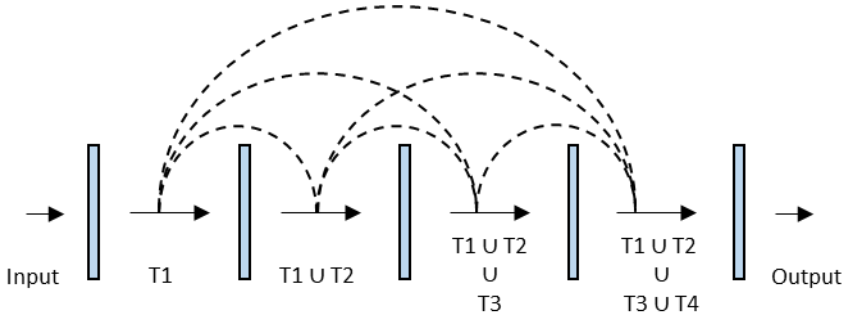


***Abbildung 11**: Schematische Darstellung eines DC-Moduls, wie es in der UNet++ Architektur vorkommt.* [55] *Der Input des Moduls erfährt wiederholte Faltungen (Faltungs-Schicht, blau markiert). Vor jeder Faltung wird der Tensor (T1 – T4) kopiert. Der Tensor T1 wird zum Tensor T2 verfaltet und anschließend erneut mit T1 verkettet.*

2.3.3 Initialisierung neuronaler Netze

Die Initialisierung eines neuronalen Netzes vor dem Training benennt die Erzeugung zufälliger Gewichte und Bias des Modells. Es umfasst keine Hyperparameter wie die Lernrate, da diese vor der Initialisierung von menschlicher Seite konfiguriert werden. Trotz der zufälligen Initialisierung liegen die Parameter innerhalb eines definierten Wertebereichs. Jedes Gewicht folgt einer von menschlicher Seite definierten Wahrscheinlichkeitsverteilung. So kann die Verteilung der generierten Parameter entweder einer Gaußschen Normalverteilung folgen oder vollständig zufällig sein. Wie auch die Eingangswerte eines neuronalen Netzes normalisiert werden müssen, damit das Netzwerk besser an das theoretische Maximum der Leistungsfähigkeit tangiert, liegen auch die Gewichte im Intervall [-1, 1]. [22, 25] Der Mittelwert liegt bei 0.

Glorot et al. konnten zeigen, dass faltende neuronale Netze besser lernen, wenn die Initialisierung einer Gauß'schen Normalverteilung mit $\sigma = 0$ folgt. Die Standardabweichung der Wahrscheinlichkeitsverteilung liegt in Abhängigkeit zu der Anzahl der Input-sowie Output-Verbindungen jeder Schicht. [50] **Formel 11** verdeutlicht dies:

$$\mu = \sqrt{\frac{2}{Kanten_{input} + Kanten_{output}}} \qquad 11$$

Im Falle des Netzwerkes von [24] trafen diese Berechnungen auch zu. Für tiefere Netzwerke wurde eine Modifikation von **Formel 12** angewendet, welche zu besseren Ergebnissen führte. [19, 49, 56]

$$\mu = \sqrt{\frac{1}{Kanten_{input} + Kanten_{output}}} \qquad 12$$

Der Unterschied beider Initialisierungsmethoden besteht lediglich im Faktor $\sqrt{2}$.

2.3.4 Aktivierungsfunktionen innerhalb neuronaler Netze

Innerhalb von künstlichen Neuronen bzw. Filtern existieren Aktivierungsfunktionen, um zu bestimmen, in welcher Form das Signal weitergeleitet wird. Nicht nur Perzeptronen und reine neuronale Netze nutzen diese Funktionen, auch faltende neuronale Netze nutzen Aktivierungsfunktionen, um Signale in faltenden Schichten zu filtern. Vor Weitergabe eines Signals an die Neuronen tieferer Schichten durchläuft das Signal eine der Aktivierungsfunktion in **Abbildung 12**.

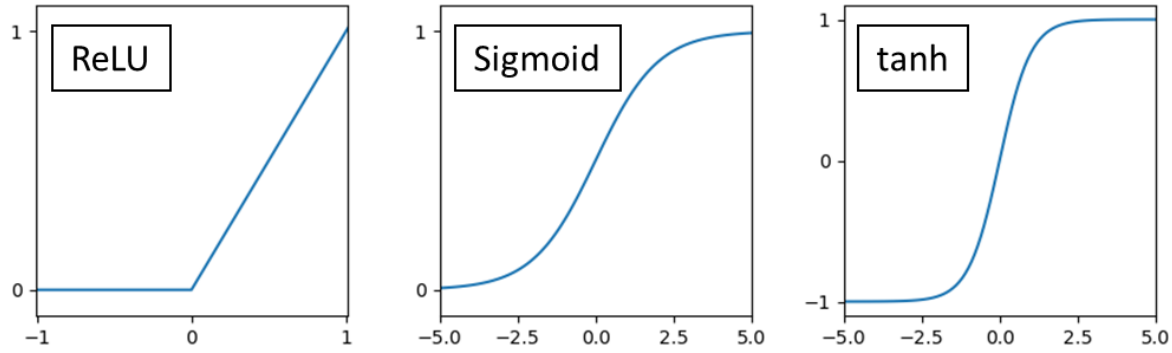


***Abbildung 12**: Verlauf gängiger Aktivierungsfunktionen.*

Ihre respektiven Gleichungen lauten:

$$ReLU: y = \text{maximum}(x, 0) \quad (13)$$

$$Sigmoid: y = \frac{1}{1 + e^{-x}} \quad (14)$$

$$tanh: y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (15)$$

Innerhalb eines neuronalen Netzes wird die ReLU-Funktion genutzt. Sie steht für die Rektifizierungsfunktion bzw. Gleichrichtungsfunktion (*Rectified Linear Unit, ReLU*). Diese zeigt ein schnelles Lernverhalten und Stabilität während des Trainings. [49] Die Sigmoid-Funktion wird in der Ausgabeschicht von klassifizierenden neuronalen Netzen und faltenden neuronalen Netzen genutzt. Da sie im Intervall [0, 1] liegt, kann sie als Wahrscheinlichkeit interpretiert werden. [24] tanh folgt der Form einer Sigmoid-Funktion, liegt aber im Intervall [-1, 1]. Sie wird in generativen Netzwerken genutzt. [47]

2.3.5 Künstliche Erweiterung der verwendeten Datensätze (Data-Augmentation)

Wird ein neuronales Netz trainiert, kann es dazu kommen, dass das Netzwerk keine diversifizierten Eigenschaften in den Trainingsdaten entdeckt und sich auf schwache, wenig ausdrucksstarke Eigenschaften des Trainingsdatensatzes fokussiert. Das Netzwerk vernachlässigt mit der Zeit allgemeine Eigenschaften und wird spezifische Eigenschaften der Trainingsdaten lernen. Darunter leidet die Performance, sei es im Bereich der Klassifizierung, Segmentierung oder Generierung von Bildern. Misst man die Leistung anhand der Kostenfunktion, wird mit zunehmendem Training eine Diskrepanz zwischen der Kosten der Trainingsdaten und den Kosten der Validierungsdaten feststellen werden können (**Abbildung 13**).

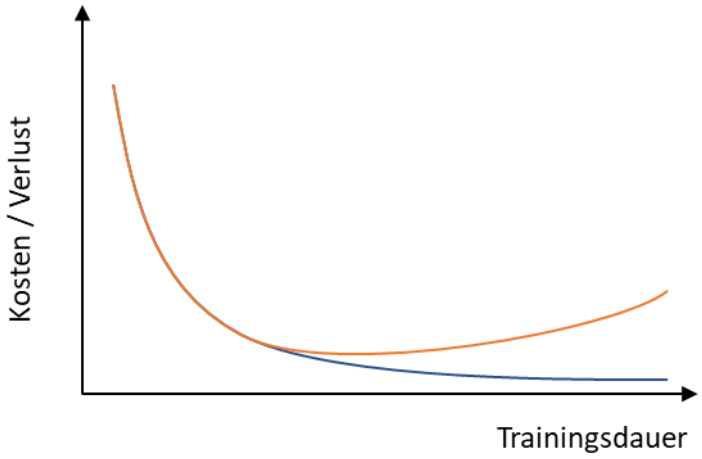


***Abbildung 13**: Schematische Darstellung der Überanpassung. Die Abbildung zeigt einen schematischen Verlauf der mittleren Kosten des Trainingsdatensatzes (Blaue Linie) gegenüber den mittleren Kosten des Validierungsdatensatzes (Orange Linie), bei zunehmender Trainingsdauer.*

Kommt es zu einer solchen Situation, lernt das Netzwerk nicht über den gesamten Eigenschafts-Raum, sondern nur über wenige spezifische Eigenschaften des Trainingsdatensatzes. Sobald diese wenigen Eigenschaften von dem System erkannt wurden, hört das Netzwerk auf zu lernen. Dies wird Überanpassung genannt und ist nicht erwünscht. Da es bei dem Training eines neuronalen Netzes also zu einer Überanpassung kommen kann, müssen geeignete Techniken genutzt werden, um das Netzwerk gegen eine Überanpassung zu schützen. Eine Möglichkeit ist die Vergrößerung des Trainingsdatensatzes. Da dies in vielen

Fällen praktisch nicht umsetzbar ist, kann dies auch künstlich mittels affiner Transformation von Bildern während des Trainings geschehen. [57] Die sequenzielle affine Transformation der unbearbeiteten Daten wird *Data Augmentation* genannt. *Data Augmentation* muss semantisch mit dem Datensatz abgestimmt werden: D. h. ein Datensatz, der stehende Menschen in diversen Posen zeigt, darf während des Trainings nicht um 180° gedreht werden, da sonst das die Eigenschaft „Menschen stehen“, verloren gehen würde. Die künstliche Vergrößerung von Datensätzen umfasst neben geometrischen Veränderungen auch das Einfügen von Rauschen oder die Änderung des Kontrastes. Somit können aus wenigen Hundert Trainingsbildern über zufällige Variation während des Trainings ad hoc unterschiedliche Bilder generiert werden. Dies hilft dem Netzwerk zu generalisieren und einer Überanpassung entgegenzuwirken. Die Bilder des Validierungsdatensatzes werden nicht verändert, um eine konstante Vergleichsgröße während des Trainings zu erhalten. Die Technik des zufälligen Veränderns ist eine Standardprozedur. Es konnte mehrfach gezeigt werden, dass die Leistung von Modellen, die auf variablen Daten trainiert wurden, sich deutlich von Modellen ohne Verwendung von *Data Augmentation* abhebt. [58, 59]

2.3.6 Leistungsbewertung neuronaler Netze

Zur kontinuierlichen Bewertung des Trainings eines neuronalen Netzes wird der verwendete Datensatz in 2 Teile gespalten. 80 % des Datensatzes werden für das Training verwendet (Trainings-Datensatz). 20 % werden für die Validierung bereitgestellt (Validierungs-Datensatz). Während die Trainingsdaten mittels Data-Augmentation möglichst variabel gestaltet werden, bleibt der Validierungsdatensatz unverändert. Diese Unterteilung erfolgt zufällig, sodass in beiden Datensätzen ein gleiches Verhältnis der Trainingsklassen vorhanden ist. Die Validierung des Modells erfolgt nach jeder Epoche.

Neben der Kostenfunktion zur Bewertung der Leistung eines neuronalen Netzes gibt es wichtige Metriken, welche auf die verschiedenen Aufgaben der Modelle abgestimmt sind. Sie leiten sich aus der folgenden Konfusionsmatrix ab:

		Wahres Ergebnis	
		Positiv	Negativ
Vorhersage des Modells	Positiv	Echt Positiv *True Positive (TP)*	Falsch Positiv *False Positive (FP)*
	Negativ	Falsch Negativ *False Negative (FN)*	Echt Negativ *True Negative (TN)*

Im Zusammenhang der Bildklassifizierung wird jedes Trainingsbild bewertet. Im Zusammenhang mit der Segmentierung ist jedes Pixel der gezeigten Bilder ein zu bewertendes Element. Die Pixel eines segmentierten Bildes ergeben summiert die Grundlage der Metrik eines Bildes.

Die Hauptmetrik zur Bewertung von Klassifizierungsnetzwerken ist die Korrektklassifizierungsrate (*Accuracy*). [60, 61] Diese gibt den relativen Anteil der richtig erkannten Beispiele an, welche definiert ist als:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \qquad 16$$

Die Hauptmetrik zur Bewertung von binären Segmentierungs-Netzwerken ist die *Intersection-over-Union* (IoU). [60]

$$IoU = \frac{TP}{TP + FP + FN} \qquad 17$$

3 Adaptierung der Methoden

3.1 Bildsegmentierung mittels UNet ++

Die Segmentierung von Bildern in zusammenhängende Bereiche aus Vorder- und Hintergrund ist eine Kernaufgabe der modernen künstlichen Intelligenz und zählt zur Klasse des überwachten Lernens. Ziel der Segmentierung ist es, einzelne Partikel (Vordergrund) klar vom Probenträger (Hintergrund) zu trennen. Überlappende Partikel werden in einer erfolgreichen Segmentierung so voneinander getrennt, dass ein schmaler Bereich aus Hintergrund zwischen diesen Partikeln eingefügt wird.

Das Ziel der Segmentierung ist über einen Zwischenschritt zu erreichen. Wie auch in der Bildklassifizierung handelt es sich bei einem Modell zur Bild-Segmentierung um ein probabilistisches Modell. Ein Segmentierungsmodell ordnet jedem Pixel des Eingangsbildes eine Wahrscheinlichkeit zu, zu Vorder- oder Hintergrund zu gehören. Im Gegensatz zur Bildklassifizierung, in welcher die Gesamtheit aller Pixel des Bildes einer Klasse zugeordnet wird, soll in der Segmentierung für jedes Pixel eine eigene Klasse identifiziert werden.

Im Anschluss zur Berechnung des Modells wird jedes Pixel, das eine Wahrscheinlichkeit von mindestens 51 % besitzt, zur Vordergrundklasse zu gehören, als Vordergrundpixel markiert. **Abbildung 14** demonstriert diesen Prozess.

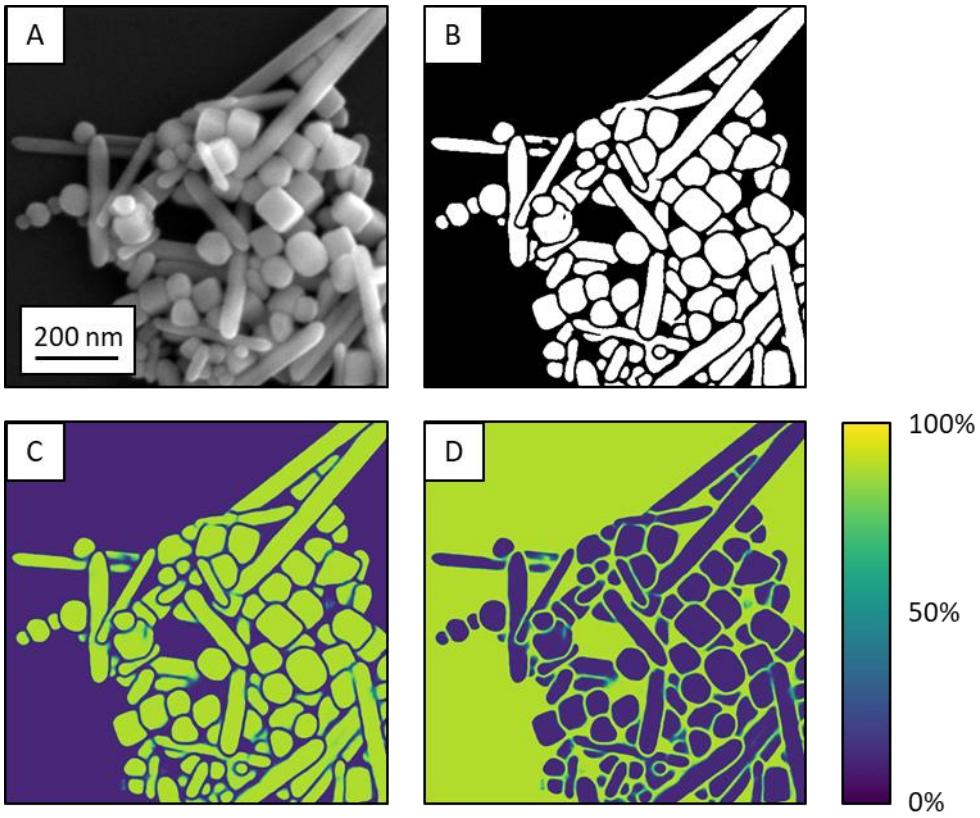


***Abbildung 14**: Bildliche Darstellung der Aufgabe eines Modells zur Segmentierung von REM-Bildern. Das Bild (**A**) wird von dem Modell in Vordergrund (weiß) und Hintergrund (schwarz) getrennt (**B**). Einzelne Partikel werden von einem schmalen Rand aus Hintergrund-Pixeln voneinander getrennt. Die Aufteilung des Bildes in eine Segmentierung erfolgt über einen Zwischenschritt. Das Modell berechnet für jedes Partikel eine Wahrscheinlichkeit, entweder zu Vordergrund (**C**) oder Hintergrund (**D**) zu gehören. Pixel, welche von dem Modell eine Wahrscheinlichkeit von mindestens 51 % erhalten zum Vordergrund zu gehören, werden als Vordergrund markiert. Pixel, welche eine geringere Wahrscheinlichkeit aufweisen, werden zu Hintergrund-Pixeln.*

Über das Verfahren der Segmentierung können zusammenhängende Vordergrundbereiche (Partikel) voneinander getrennt werden. Jedes Partikel kann im Folgenden bezüglich seiner Position und Ausdehnung vermessen werden. Dadurch, dass Partikel so getrennt werden können, bietet die Segmentierung eine Grundlage für weitere Analysen. Die Segmentierung ist ein wichtiger Schritt in der automatischen Analyse von REM-Bildern. Nur wenn Partikel in diesem Schritt voneinander getrennt wurden, können Morphologie und Größe des Partikels ausreichend genau bestimmt werden.

Neben dem Gebrauch im Bereich der Materialwissenschaften findet sich auch in der Medizin eine breite Anwendung für Segmentierungs-Modelle. [6, 12, 16] Nicht selten übertreffen KI-Systeme dabei die menschliche Fähigkeit zur Erkennung von Objekten in REM-Aufnahmen. [16]

Für die hier vorliegende Arbeit wurden 4 eigenständige Netzwerke mittels 2 unterschiedlicher Datensätze trainiert. Ein Datensatz enthielt nur STEM-Aufnahmen, ein weiterer nur SE-Aufnahmen. Dies war erforderlich, da sich SE- und STEM-Aufnahmen durch ihre Erscheinung so weit voneinander unterschieden, dass ein einzelnes Modell nicht in der Lage war, sich an beide Bildtypen gleichzeitig anzupassen. Alle trainierten Netzwerke basierten auf der UNet++ Architektur. [55] Des Weiteren wurden für beide Datensätze unterschiedliche Gewichtungskarten verwendet, welche den Netzwerken halfen, eng beieinanderliegende Partikel aufzutrennen.

3.1.1 Architektur

Das hier verwendete Modell zur Segmentierung von REM-Bildern basiert auf der UNet++ Architektur. [55, 62] Es gehört zur Klasse der *Convolutional-Autoencoder* (CAE). [63] CAE sind in der Lage, die Eigenschaften eines Bildes zu erfassen und daraus ein neues Bild zu generieren. Die Pixelwerte des generierten Bildes sind dabei direkt vom Inhalt des Eingangsbildes abhängig. So ist es möglich, durch Paare aus REM-Bildern und deren händisch erstellten Annotationen ein Modell zu trainieren, das eigenständig in der Lage ist, REM-Bilder zu segmentieren. **Abbildung 15** zeigt die verwendete UNet++ Architektur.

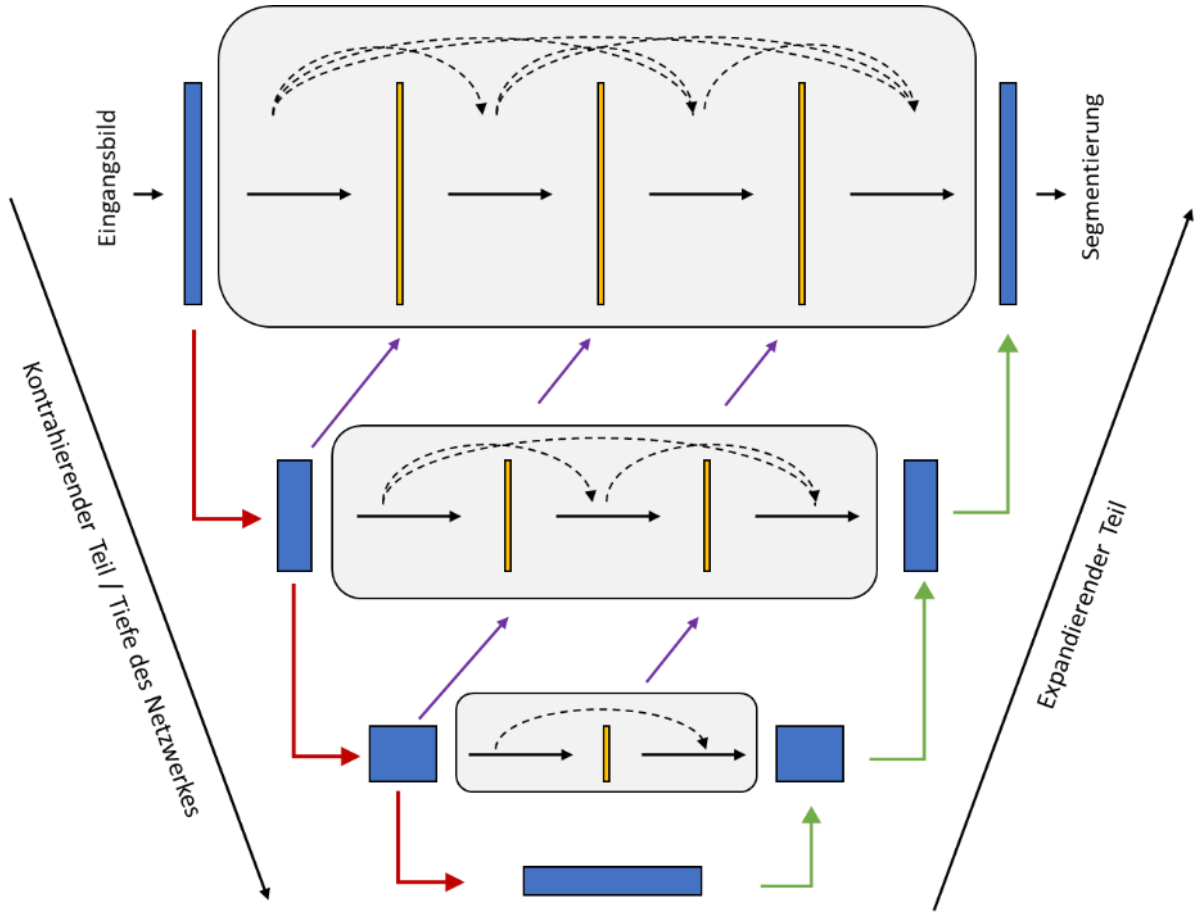


Abbildung 15: *Die UNet++ Architektur ist eine komplexe Weiterentwicklung der UNet Architektur.* [56] *Der kontrahierende Teil der Architektur nutzt eine wiederholte Abfolge von Faltungs- und MaxPool Schichten (rote Pfeile). Die Anzahl der Filter wird nach jeder MaxPool Schicht verdoppelt, wodurch die Tiefe des Eigenschafts-Tensors (blaue Boxen) zunimmt. Durch diesen Prozess wird der Bildinhalt stark kodiert. Nach den Faltungen und vor der MaxPool-Operation, wird der Eigenschafts-Tensor kopiert. Die kopierten Eigenschafts-Tensoren werden mittels dichter Faltungsmodule weiter verfaltet (graue Kästen). Innerhalb der Module befinden sich erneut Faltungs-Schichten (gelb). Die dichten Faltungsmodule sind untereinander ebenfalls vernetzt (lila Pfeile). Durch das stetige Kopieren der Eigenschafts-Tensoren und die Übergabe an höhere Schichten des Netzwerkes ergibt sich eine lineare Abhängigkeit höherer Schichten zu tieferen Schichten. Diese lineare Abhängigkeit ermöglicht es, die Parameter des Netzwerkes zu minimieren, da die Parameter der höheren Schichten von einer Vielzahl unterschiedlicher Eigenschaften tieferer Schichten abhängig gemacht werden. Im expandierenden Teil des Netzwerkes wird aus dem Eigenschafts-Tensor schrittweise wieder ein Bild geformt. Hierfür werden entfaltende Schichten eingesetzt (grüne Pfeile). Während der Entfaltung werden die zuvor kopierten Eigenschafts-Tensoren mit den entfalteten*

Eigenschafts-Tensoren verkettet. Ronneberger et al. [56] konnten zeigen, dass das Konkatenieren des Eigenschafts-Tensors vom kontrahierenden Teil mit dem des expandierenden Teils das Netzwerk schneller zu trainieren lässt. Zudem werden die Kanten der zu segmentierenden Objekte besser erfasst. Hierdurch entsteht allerdings eine „semantische Lücke". Die noch schwach verfalteten Eigenschafts-Tensoren des kontrahierenden Teils werden gleichwertig neben die stark verfalteten Eigenschafts-Tensoren des expandierenden Teils gestellt. Die dichten Faltungsmodule sind in der Lage, die semantische Lücke zu schließen. Das Netzwerk lernt dadurch schneller und benötigt weniger Parameter im Training, um die gleiche Leistung zu erzielen wie die ursprüngliche UNet Architektur. [55]

3.1.2 Datensatz von STEM-Bildern

Für das Training des neuronalen Netzwerkes, das STEM-Bilder segmentiert, wurde ein Datensatz von 19 unterschiedlichen Bildern verwendet. Die verwendeten Bilder zeigen Nanopartikel unterschiedlicher Proben. Die Partikelsysteme bestehen aus Au und Ag mit diverser Morphologie. Die Proben, von denen die Bilder stammen, wurden von unterschiedlichen Personen mittels unterschiedlicher Synthesen hergestellt. Aufgrund der 2-dimensionalen Darstellung von STEM-Aufnahmen wurden die Partikelformen nach 2-dimensionalen Objekten benannt. [64] Die Aufnahmen zeigen Kreise & Ellipsen, Vierecke & Quadrate, hexagonale & trigonale Plättchen sowie pentagonale Bipyramiden & Oktaeder. Weiterhin sind Agglomerate der genannten Formen vorhanden. Agglomerate wurden als nicht trennbare Überlappungen von einzelnen Partikeln definiert.

Im Falle von STEM-Bildern können 2 primäre Streuwinkel von Elektronen zur Bilderzeugung genutzt werden, welche sich durch ihren Winkel der Ablenkung beim Durchqueren der Probe unterscheiden. Hellfeld-Bilder (HF) werden durch Detektion der Elektronen generiert, die eine geringe Ablenkung erfahren. Der Detektor liegt entlang der optischen Achse des Elektronenstrahls. Dieser detektiert die direkte Elektroneneinstrahlung, womit der Kontrast von der Dicke der durchstrahlten Probe anhängt. Elektronen, die in hohen Winkeln von der Probe abgelenkt werden, werden mittels eines konzentrischen Detektors registriert, der die optische Achse umgibt. Dieser wird *high angle annular dark field (HAADF)* Detektor bezeichnet. Daraus wird ein Bild generiert, dessen Kontrast von der Ordnungszahl der in der

Probe enthaltenden Elemente abhängig ist. Je höher die Ordnungszahl, desto heller erscheint das Bild an dieser Stelle. [65]

Für das Training wurden Hellfeld-Bilder linear invertiert, um Partikeln eine helle Färbung zu geben und Hintergrund eine dunkle (**Abbildung 16**). Dies war nötig, damit die KI einheitliche Daten erhält. Die Invertierung erfolgte nach der nachstehenden Formel:

$$HAADF_{invertiert} = |HF - 2^n| \qquad 18$$

n steht für die Tiefe des Bildes in Bit. 2^nist die Anzahl der maximal möglichen Graustufen. **Abbildung 16** zeigt eine solche Invertierung für eine HF-Aufnahme von Au-Nanopartikeln:

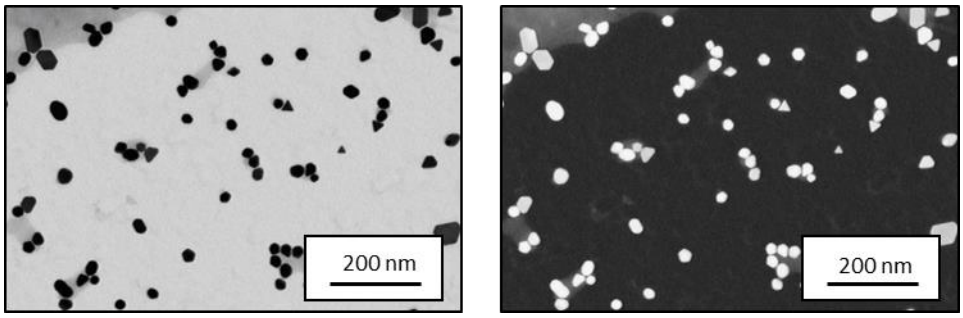


***Abbildung 16**: Die Abbildung zeigt eine HF-Aufnahme (links) sowie die linear invertierte Form des Bildes.*

Für jedes REM-Bild des Datensatzes wurde händisch eine binäre Klassifizierungs- bzw. Annotationsmaske (*Annotation Masks*) erstellt. Die Klassifizierungsmaske teilte jedem Pixel die Klasse Vordergrund oder Hintergrund zu. Um Partikel, welche nahe beieinander liegen, trennen zu können, wurde jedes Partikel von einer schmalen Linie von Hintergrund-Pixeln umgeben. Überlappende Partikel, welche auch von einem Menschen nicht trennbar sind, wurden gemeinsam von einer Linie aus Hintergrund umgeben. Dieser Prozess wurde mittels der Foto-Bearbeitungssoftware GIMP 2.10.12 durchgeführt. [66] Im Anschluss ist der Datensatz zufällig in 12 Trainingsbilder sowie 7 Validierungsbilder aufgeteilt worden.

3.1.3 Datensatz von SE-Bildern

Der Datensatz von SE-Bildern bestand aus 52 Einzelaufnahmen. Die Einzelbilder enthielten Partikel unterschiedlicher elementarer Zusammensetzung und Morphologie. Es wurde bei der Erstellung des Datensatzes darauf geachtet, dass eine möglichst hohe Diversität verschiedener

Formen erreicht wurde. Da es sich bei SE-Aufnahmen um pseudo 3-dimensionale Aufnahmen handelt, wurden die enthaltenen Formen nach ihrem 3-dimensionalen Pendant benannt. Verwendet wurden Nanopartikel mit folgender Morphologie: Kugeln, Stäbchen mit quadratischem, hexagonalem und rundem Querschnitt, Würfel & abgestumpfte Würfel, Tetraeder, Pyramiden, Oktaeder und Partikel undefinierbarer Form. Die Proben einhielten Partikel folgender elementarer Zusammensetzungen: SiO_2, TiO_2, ZnO, Calciumphosphat sowie Au und Ag. Wie auch schon die Bilder des STEM-Datensatzes wurden die Partikel von mehreren Personen mittels unterschiedlicher Synthesen hergestellt.

SE-Aufnahmen stellen pseudo 3-dimensionale Aufnahmen von Nanopartikeln dar. Abgebildete Partikel können sich berühren oder überlappen. Um sich überlappende Partikel trennen zu können, musste in der binären Klassifizierungsmaske jedes Partikel von einem schmalen Rand aus Hintergrund umgeben werden. Bei der Erstellung der Klassifizierungsmasken wurde daher darauf geachtet, dass eine solche Linie außen um das Partikel gelegt ist. Die Breite der Umrandung betrug ca. 1-3 Pixel.

Die binären Klassifizierungsmasken wurden händisch für jedes Bild mittels der Foto-Bearbeitungssoftware GIMP 2.10.12 hergestellt. [66] Wie auch für STEM-Bilder, wurde der Datensatz im Anschluss zufällig aufgeteilt, dass in 30 Trainingsbilder sowie 12 Validierungsbilder vorlagen. Zusätzlich wurden 32 kleinere Bilder von agglomerierten TiO_2 Nanopartikeln in den Trainingsdatensatz und 8 Bilder gleichen Inhaltes in den Validierungsdatensatz aufgenommen, welche zuvor von Rühle et al. veröffentlicht wurden. [67] Dies diente dazu, dem Datensatz eine höhere Variabilität zu geben.

3.1.4 Gewichtungskarten für SE-Bilder

Das verwendete Modell wurde nach dem Gradientenabtiegsverfahren trainiert. Zunächst berechnete das Modell eine Wahrscheinlichkeitsverteilung für jedes Pixel zur Klasse Hintergrund oder Vordergrund zu gehören. Über die Kostenfunktion wurde nun die Abweichung jedes Pixels von der wahren Klasse berechnet. Anschließend wurde hieraus die mittlere Abweichung aller Pixel berechnet. Dies waren die mittleren Kosten eines Bildes. Die mittleren Kosten bildeten die Grundlage für die Anpassung der Parameter des Modells. In diesen Prozess griff die Gewichtungskarte ein. Wenn die mittleren Kosten des Bildes

berechnet wurden, ist jedes Pixel nach dem korrespondierenden Wert der Gewichtungskarte modifiziert worden (**Abbildung 17**). Somit konnten individuellen Pixeln höhere Werte beigemessen werden. Die Wertung erfolgt dabei anhand der Entfernung eines Pixels von den nächsten 2 benachbarten Partikeln.

Das Training einer KI für die Segmentierung von SE-Bildern ist fehleranfälliger, je kleiner die Bereiche sind, welche segmentiert werden sollen. Dies schließt Vordergrundobjekte wie auch Hintergrund ein. Kleinere Bereiche erfahren unabhängig ihrer Form eine durchschnittlich schlechtere Segmentierung als große Flächen. [68] Ronneberger et al. stellten Gewichtungskarten vor, mit welchen das Modell auf Bildbereiche oder sogar einzelne Pixel fokussiert wurde, welche nahe an 2 Kanten von Vordergrundobjekten liegen. [56] Dies diente der Trennung von sich berührenden Vordergrundobjekten, welche in der händisch erstellten Klassifizierungsmaske lediglich durch eine schmale Linie aus Hintergrund voneinander trennbar sind. Das Netzwerk lernte dadurch verstärkt die Ränder von Objekten zu identifizieren. Die eingeführten Gewichtungskarten sind 2-dimensionale Matrizen mit der gleichen Größe wie die Klassifizierungsmaske. In dieser Arbeit wurde Hintergrundpixeln mittels der Gewichtungskarte eine höhere Relevanz im Training zuordnet, wenn diese Pixel nahe an Partikeln lagen. Ihre Berechnung erfolgt anhand der händisch erstellten Klassifizierungsmaske, wie in **Formel 19** dargestellt.

$$w_{dist} = w_0 * \exp\left(-\frac{\big(d_1(x) + d_2(x)\big)^2}{2\sigma^2}\right) + 1 \qquad 19$$

Die Werte der Gewichtungskarte w_{dist} sind abhängig von der Distanz $d_1(x)$ & $d_2(x)$ des jeweiligen Pixels zu den zwei nächsten benachbarten Partikelgrenzen. Die Stärke der Gewichtung der Pixel wird über zwei Hyperparameter w_0 & σ bestimmt. Experimente zeigten gute Segmentierungsergebnisse bei $w_0 = 10$ und $\sigma = 7$. Das Maximum des Wertebereichs für Pixelgewichte wird über w_0 definiert und beträgt in diesem Fall 10. Das Minimum liegt bei null. Da eine Gewichtung von null dazu führt, dass die Kostenfunktion für dieses Pixel einen Wert von null liefert, würden diese Pixel keinen Einfluss mehr auf die Parameteroptimierung haben. Dies ist dem Prozess der Parameteroptimierung hinderlich. Daher wird jedem Pixel ein Wert von 1 addiert.

Die Grundlage der Gewichtungskarten bildet im Falle von SE-Bildern ausschließlich die Klassifizierungsmaske und die sich daraus ergebende Distanz von Hintergrundpixeln zu den

Partikeln. Dieser Ansatz soll im Folgenden distanzbasierte Gewichtungskarte genannt werden. Im Falle von STEM-Bildern wurden Gewichtungskarten weiter modifiziert.

3.1.5 Gewichtungskarten für STEM-Bilder

Für die hier vorliegende Arbeit wurde eine Variation der distanzbasierten Gewichtungskarten eingeführt.

Während in SE-Bildern Partikel pseudo 3-dimensional abgebildet sind, ist auf STEM-Bildern eine 2-dimensionale Repräsentation der Partikel zu sehen. Wenn Partikel übereinander liegen, können sie zusammenhängende, von einem Menschen nicht trennbare Agglomerate formen. Diese Agglomerate liefern keine visuellen Anzeichen, wie viele Partikel das Agglomerat bilden oder welche Form diese Partikel besitzen. Eine Trennung von sich berührenden Partikeln ist dagegen begrenzt möglich. Liegen Partikel nahe beieinander, nimmt die Intensität (Grauwerte) der Pixel zwischen den Partikeln mit abnehmender Distanz der Partikel zu. Pixel zwischen 2 Partikeln können so hell werden, dass beide Partikel miteinander „verschmelzen“. Um das Modell darauf zu trainieren, sich berührende Partikel zu trennen, wurden die Gewichtungskarten so modifiziert, dass die Intensität des Bildes in die Gewichtung einfloss. Theoretisch sollte diese Modifizierung dazu führen, dass sich das Modell während des Trainings darauf fokussiert, diese Partikel zu trennen. Für diese Arbeit wurde **Formel 19** wie nachfolgend in **Formel 20** modifiziert.

$$w_{int} = w_{dist} + \gamma * I_x \qquad 20$$

Um die Intensität eines STEM-Bildes im Training zu berücksichtigen, wurde die distanzbasierte Gewichtungskarte w_{dist} modifiziert. Das Bild wird zunächst auf das Intervall [0, 1] normalisiert. Im Anschluss wurde das normalisierte Bild I_x mit dem Term γ gewichtet und Pixel für Pixel mit der distanzbasierten Gewichtungskarte addiert. Der Gewichtungsterm γ kontrolliert die Stärke des Einflusses der Bildintensität auf die finale Gewichtungskarte. Experimente zeigten einen optimalen Wert von $\gamma = 2$. Somit können STEM-Gewichtungskarten Werte zwischen 1 und 13 annehmen. Aufgrund des Einflusses der Bildintensität auf die Gewichtungskarten soll diese Variation als intensitätsbasierte Gewichtungskarten bezeichnet werden.

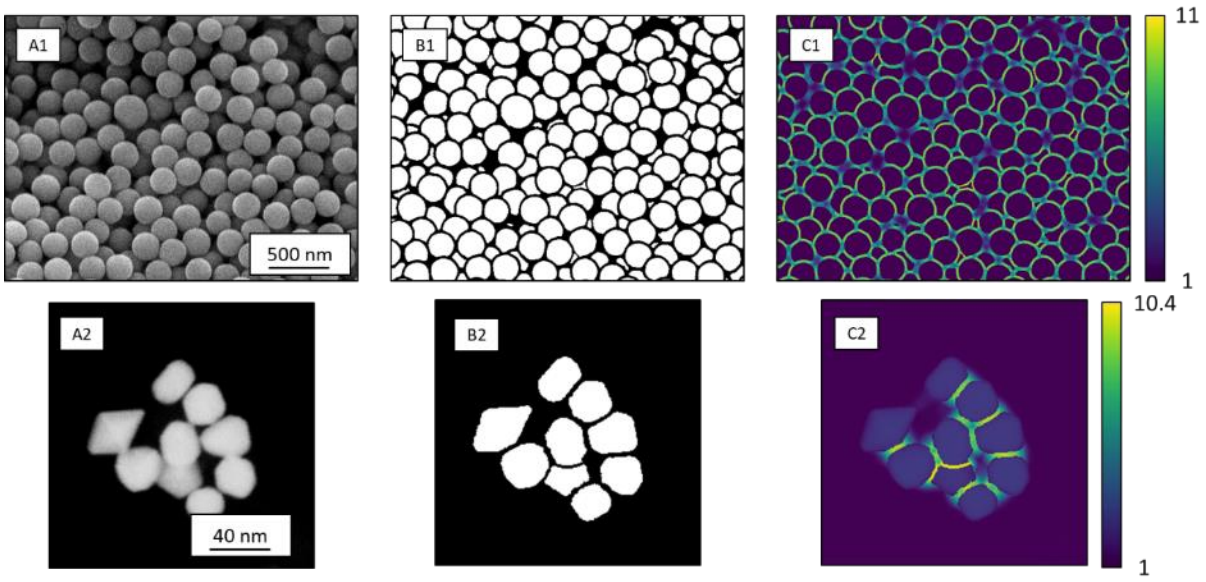


***Abbildung 17**: Gewichtungskarten für SE- und STEM-Bilder. Es sind SiO_2 Nanopartikel gezeigt, die im SE-Modus aufgenommen wurden. (**A1**) Gezeigt ist die händisch erstellte Annotation (**B1**). Die distanzbasierte Gewichtungskarte (**C1**) wurde nach **Formel 19** berechnet. Zudem ist ein Ausschnitt eines STEM-Bildes von Au Nanopartikeln (**A1**) mit zugehöriger Segmentierung gezeigt (**B2**). Die Gewichtungskarte des STEM-Bildes (**C2**) wurde nach **Formel 20** berechnet. Für eine bessere Sichtbarkeit wurden die Farbskalen an den Bildinhalt angepasst.*

3.1.6 Training

Das Training der UNet++ Modelle fand mittels Gradientenabstiegsverfahren statt. Zur Validierung der intensitätsbasierten Gewichtungskarten wurden 4 UNet++ Modelle trainiert, 2 für STEM- und 2 für SE-Bilder. Für jeden Datensatz wurde ein Modell trainiert, welches intensitätsbasierte Gewichtungskarten nutzte und ein Modell, welches distanzbasierte Gewichtungskarten nutzte. Dies diente der Prüfung der jeweiligen Gewichtungsmethode und der potenziellen Qualitätsverbesserung für STEM- und SE-Bilder.

Alle 4 Netzwerke wurden identisch nach He et al. initialisiert [49] und nutzten die Kreuzentropie aus Kostenfunktion (**Formel 2**). [43] Die Größe der Bilder, die mittels der verwendeten Architektur segmentiert wurde, betrug 512x512 Pixel. Die Größe wurde durch den Speicherbedarf des Modells limitiert, der mit zunehmender Größe der Bilder anstieg. Die Bilder des Datensatzes waren unterschiedlich groß, lagen aber durchschnittlich bei 1000x1500 Pixeln.

Um während des Trainings eine hohe Variabilität der Daten gewährleisten zu können, wurden die Bilder vor jeder Epoche neu geladen, auf das Intervall [0, 1] normalisiert und augmentiert. Die *Data-Augmentation* enthielt die folgenden Schritte, welche für jedes Bild zufällig ausgeführt wurden. **Abbildung 18** fasst diesen Prozess zusammen.

i. Drehen um bis zu 360°
ii. Horizontales und vertikales Spiegeln des Bildes
iii. Bis zu 15 % Vergrößerung des zentralen Bildbereichs und Skalierung auf ursprüngliche Größe
iv. Scherung der Bildkanten mit bis zu 15 % der Bildhöhe
v. Addition von Gauß‘schem Rauschen mit $\sigma = 0$ und $\mu = 5$ für jedes Pixel
vi. Normalisierung des Bildes auf das Intervall [0, 1]

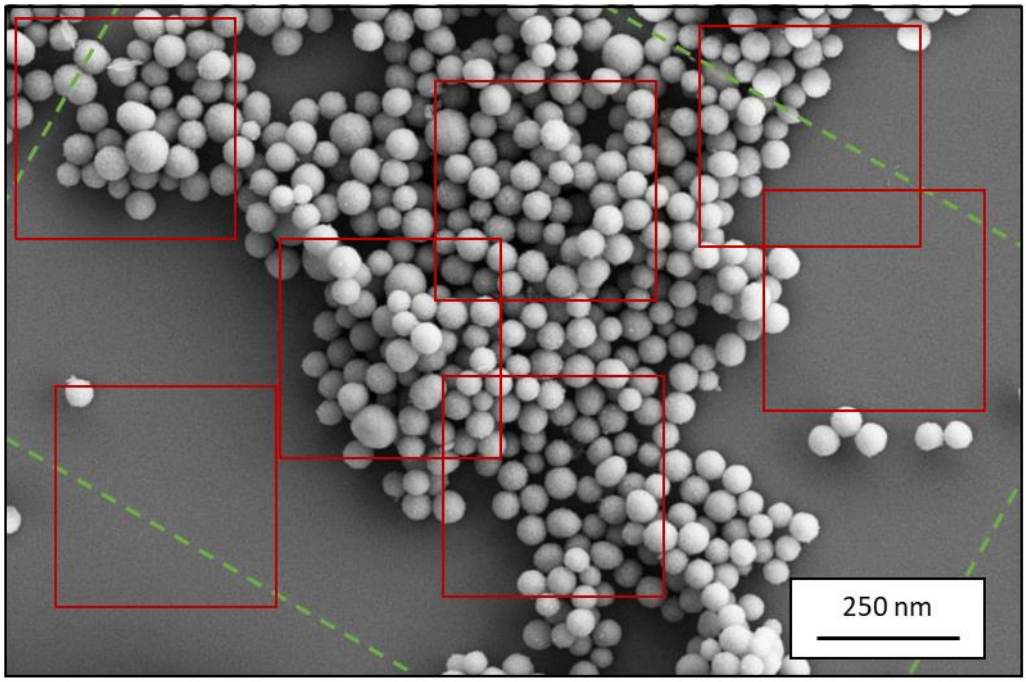


***Abbildung 18**: Schematischer Prozess der künstlichen Bildveränderung und anschließendem Ausschneiden der Patches. Das gezeigte Bild wurde zufällig gedreht. Um leere Bereiche, welche durch die Drehung entstanden sind, aufzufüllen, wurde das Bild entlang der Bildkanten (grün gestrichelte Linie) nach außen gespiegelt. Es ergibt sich ein geschlossenes Bild. 512x512 Pixel große Patches (rote Quadrate) wurden zufällig ausgeschnitten und dem Datensatz hinzugefügt. Da für das Training von jedem Bild auch die Segmentierung und die jeweilige Gewichtungskarte benötigt wurden, erfuhren diese die gleichen Veränderungen wie das Bild. Es wurden ebenfalls die gleichen Patches ausgeschnitten.*

Die Variation des Datensatzes half dem Netzwerk zu lernen. Die vorliegenden Bilder wurden durch die Data-Augmentation nicht in ihrer Größe beeinflusst. Daher sind diese zu groß für das Netzwerk. Für das Training der Netzwerke wurden nach dem Prozess der Data-Augmentation aus jedem Bild zufällig Bildbereiche (*Patches*) mit einer Größe von 512x512 Pixeln ausgeschnitten. Der Prozess aus (i) Laden der Bilder, (ii) Data-Augmentation und (iii) zufälligem Ausschneiden von Patches wurde in jeder Epoche wiederholt. Anschließend wurde das Netzwerk mit den so generierten Patches trainiert. Pro Epoche wurden 450 Patches generiert. Die Anzahl der Patches, welche pro Bild ausgeschnitten werden, hing von der originalen Größe des Bildes ab. Größere Bilder wurden häufiger beprobt als kleinere. Dieser Prozess ist zeitaufwendiger als eine einmalige Beprobung der Bilder, führte aber auch zu einer

deutlichen Variation der Daten. Eine Variation von Daten half dem Netzwerk, sich an viele mögliche Partikelvariationen anzupassen.

Die Parameter des Modells wurden nach jedem Bild optimiert. Hieraus resultierte eine Mini-Batch-Größe von 1. Eine Epoche endete, wenn alle Patches verwendet wurden. Die Modelle wurden so trainiert, dass das Training endete, sobald für 20 Epochen keine Verbesserung der Leistung mehr erbracht werden konnte. Die Leistung des Netzwerkes wurde mit den immer gleichen Patches des Validierungsdatensatzes gemessen. Der Validierungsdatensatz blieb statisch. Das heißt, es wurden die gleichen Patches aus Bildern generiert, um alle Netzwerke zu validieren. STEM- und SE-Datensatz erhielten jeweils einen eigenen Validierungsdatensatz. Bilder des Validierungsdatensatzes erfuhren keine Veränderung durch Drehung, Zoom, Rauschen etc., um vergleichbare Ergebnisse zu erhalten.

3.2 Partikelklassifizierung mittels AlexNet (STEM)

3.2.1 Zielsetzung

Die Klassifizierung von Nanopartikeln in Bezug auf ihre Form wurde mittels AlexNet erreicht. Der Name leitet sich aus dem Namen des Entwicklers (Alex Krizhevsky) ab. [24] Das Ziel der Anwendung von AlexNet ist es, die Form des auf einem Bild gezeigten Nanopartikels zu bestimmen. AlexNet wurde darauf trainiert, zwischen unterschiedlichen, typischen Morphologien zu unterscheiden. Die Korrektklassifizierungsrate sollte gleich oder höher einer menschlichen Referenzgruppe liegen. Die angestrebte Korrektklassifizierungsrate lag bei mindestens 75 %. Dies entsprach in einer Umfrage der mittleren Wahrscheinlichkeit, mit welcher eine Gruppe aus erfahrenen menschlichen Analysten ein Partikel einer der zuvor definierten Klassen zuordnen würde. [69] Für diese Analyse der Partikel müssen die Nanopartikel als Einzelbilder vorliegen. Sind mehrere Partikel auf einem Bild sichtbar, ist mit dem hier verwendeten Modell keine Klassenzuordnung möglich. Es wurden einzelne Partikel durch das Ausschneiden von Vordergrundobjekten erhalten.

3.2.2 Architektur

Die verwendete Architektur basiert auf der Architektur von Zeiler & Fergus. [27] Sie besteht aus aufeinander aufbauenden Faltungs- und MaxPool-Schichten, welche in vollvernetzten Schichten enden. Die letztendliche Ausgabe der vollvernetzten Schichten liefert eine Wahrscheinlichkeitsverteilung, in welcher die Wahrscheinlichkeit des Bildes notiert wird, zu einer der trainierten Klassen zu gehören. Das Modell nutzt zudem mehrere Dropout-Schichten. [24] Dropout ist eine effektive Maßnahme, um ein Modell davor zu bewahren, sich zu sehr an spezifische Eigenschaften des Trainingsdatensatzes anzupassen. In einer Dropout-Schicht wird ein Teil der Neuronen zufällig in jedem Anpassungsschritt (nach jedem Batch) des Netzwerkes ausgestellt. Diese Neuronen tragen in diesem Schritt nicht zur Entscheidung des Netzwerkes bei. Dadurch wird das Netzwerk gezwungen, sich in gleichem Maße auf alle Neuronen in der Klassifizierung zu verlassen. **Abbildung 19** zeigt den Aufbau der Architektur.

Durch Betrachtung der Auslastung der einzelnen Schichten konnte im Vergleich zur ursprünglichen Architektur von Zeiler & Fergus [27] die Parameteranzahl reduziert werden. Die

resultierte in einem schnelleren Training und Anwendung. Besonders im Hinblick auf die Auswertung von Bildern auf durchschnittlichen Computern ist dies von Vorteil.

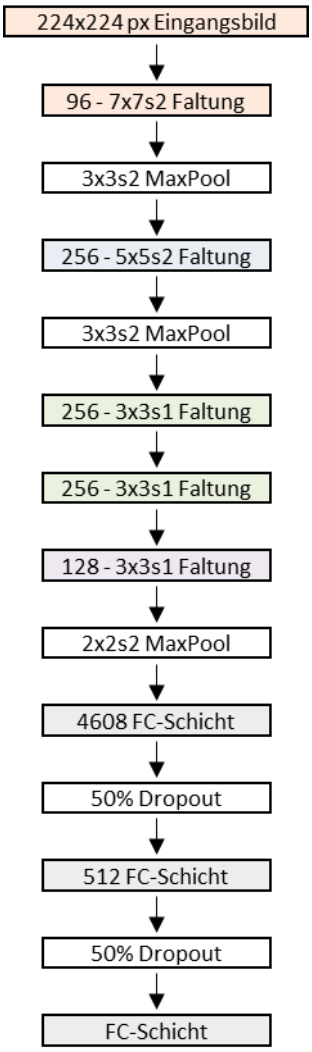


***Abbildung 19**: Architektur von AlexNet, nach den für diese Arbeit vorgenommenen Modifizierungen. Das Model besteht aus einer Reihe von Faltungsschichten und MaxPool Schichten, um die Informationen des Bildes zu Extrahieren. Gleichartige Faltungsschichten sind in identischen Farben dargestellt. Zusätzlich zur Filtergröße entscheidet die Schrittweite s, mit welcher ein Filter das Bild rastert, über die identifizierbaren Bild-Eigenschaften. In diesem Modell beträgt die Schrittweite der ersten beiden Faltungs-Schichten $s = 2$, mit großen Filtern von 7x7 und 5x5 px. Dies führt zu einer groben Abtastung des Bildes. Im Falle von Nanopartikeln im STEM Modus, die kaum Morphologische Eigenschaften aufweisen, ist eine derartige Abtastung ausreichend. Das Modell besitzt in der ersten Schicht 96 Filter und in den darauffolgenden 128 bis 256 Filter. Nach der letzten MaxPool Schicht wird der Eigenschaftstensor, durch Konkatenieren aller Werte, zu einem Vektor umgeformt und, mittels 2 vollvernetzter (FC) Schichten, in eine der 8 Klassen eingeteilt. Das Model berechnet hierfür die Wahrscheinlichkeit des Bildes zu einer der möglichen Klassen zu gehören. Um aus einer Wahrscheinlichkeitsverteilung eine definierte Klasse zu bestimmen, wird die Klasse ausgewählt, in welcher das Partikel die höchste Wahrscheinlichkeit aufweist.*

3.2.3 Datensatz

Der Datensatz der STEM-Bilder enthielt insgesamt 7125 Bilder und wurde eigens für diese Arbeit erstellt. Da bei einer Anzahl von ca. 3000 Bildern keine Verbesserung der Leistung des Netzwerkes mehr stattfand, wurde der Datensatz im Training auf ca. 3000 Bilder reduziert. Dies diente einzig einem schnelleren Training. Der Datensatz wurde in einen Trainingsdatensatz (80 %, ca. 2400 Bilder) und einen Validierungsdatensatz (20 %, ca. 600 Bilder) aufgeteilt. Die Aufteilung erfolgte automatisch. Die im Datensatz enthaltenden Partikel bestanden aus Ag und Au.

Für die Erstellung des Datensatzes wurden Partikel über eine semiautomatische Routine gefunden. Einige 100 vollständige STEM-Aufnahmen wurden mittels trainiertem Segmentierungs-Modell in einzelne Partikel und Hintergrund getrennt. Die Partikel wurden ausgeschnitten und gespeichert. Dadurch wurden einigen 10.000 Partikel gewonnen. Diese Partikel wurden dann manuell nach ihren Formen eingeteilt. Zudem erfolgte eine Qualitätskontrolle der Partikelformen. War keine Form zu erkennen, wurden die Partikel verworfen. Anhand der in den Bildern präsenten Partikelformen wurden zunächst Klassen definiert. Die Klassennamen entsprachen den vorliegenden Partikelformen. Entsprechend der Empfehlung von Muñoz-Mármol wurden die Partikelklassen nach der 2-dimensionalen Repräsentation der erwarteten 3-dimensionalen Form benannt. [64] Das Netzwerk wurde darauf trainiert zwischen Kugeln, Stäbchen, Dreiecken, Quadraten, Fünfecken, Sechsecken, Agglomeraten und abgeschnittenen Partikeln bzw. Hintergrund zu unterscheiden.

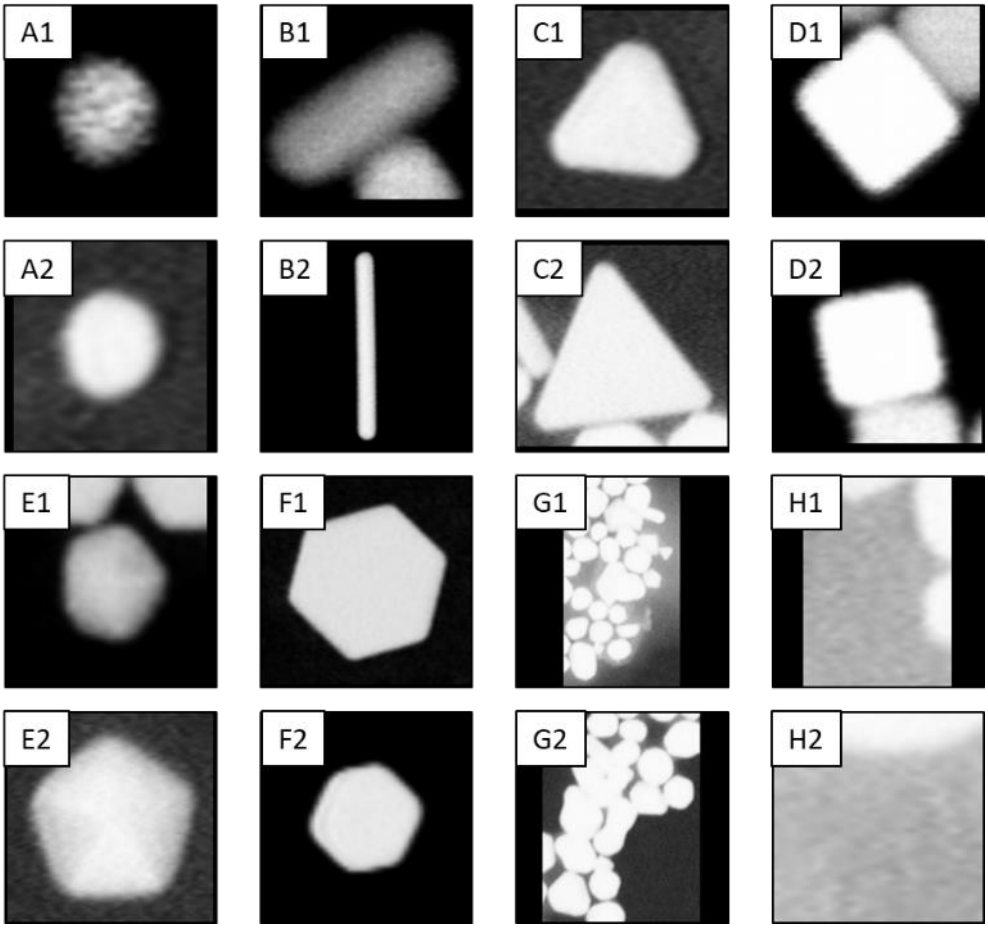


***Abbildung 20**: Typische Partikel aus STEM-Aufnahmen, die im Training verwendet wurden. Kugeln (**A**), Stäbchen (**B**), Dreiecke (**C**), Quadrate (**D**), Fünfecke (**E**), Sechsecke (**F**), Agglomerate (**G**) und verdeckte Partikel und Hintergrund (**H**). Das Netzwerk konnte lediglich quadratische Aufnahmen einer Größe von 224x224 Pixeln verarbeiten. Die aus STEM-Aufnahmen ausgeschnittenen Partikel entsprechen nicht immer dieser Größe und waren zudem nicht immer quadratisch. Um das Seitenverhältnis nicht zu verzerren, wird die lange Seite der Ausschnitte auf 224 Pixel skaliert und dann mit zusätzlichen Pixeln aus Nullwerten auf eine Größe von 224x224 Pixeln aufgefüllt* (Zero-Padding*).*

3.2.4 Training

Das Training erfolgte mittels Gradientenabstiegsverfahren. Das Netzwerk wurde nach He et al. initialisiert [49] und nutzte die Kreuzentropie als Kostenfunktion (**Formel 2**). [43] Das Training erfolgte mittels eines statischen Datensatzes. Die Bilder wurden vor dem Training zufällig in Trainingsbilder und Validierungsbilder aufgeteilt.

Während des Trainings wurden die Trainingsbilder augmentiert, um eine Variation der Daten zu bewirken. Die Data-Augmentation beinhaltete folgende zufällige Schritte:

i. Drehung um bis zu 360°
ii. Horizontales & vertikales Spiegeln des Bildes
iii. Bis zu 15 % Zoom
iv. Translation/Scherung der Bildkanten mit bis zu 15 % der Kantenlänge
v. Ausschneiden eines Bildbereiches mit 90 % der Kantenlänge des Originalbildes
vi. Addition von Gauß‘schem Rauschen mit $\sigma = 0$ und $\mu = 10$ für jedes Pixel
vii. Normalisierung auf das Intervall [-1, 1]

Nach der Augmentation wurden je 64 Bilder zu einem Mini-Batch vereinigt. Die Parameteroptimierung des Modells erfolgte nach dem Zeigen jedes Mini-Batches. Nachdem alle Mini-Batches des Trainingsdatensatzes gezeigt wurden, endete eine Epoche und das Model wurde validiert. Die Validierung erfolgte mit allen unveränderten Bildern des Validierungsdatensatzes. Unverändert bedeutet, dass diese lediglich normalisiert wurden. Im Anschluss wurden die Trainingsbilder neu augmentiert und eine neue Epoche begann. Die Reihenfolge, in welcher die Bilder pro Epoche dem Netzwerk gezeigt wurden, variierte ebenfalls. Das Training wurde beendet, nachdem in 20 aufeinanderfolgenden Epochen keine Verbesserung in der Erkennung von Validierungsbildern mehr stattfand.

3.3 Partikelklassifizierung mittels ResNet34 (SE)

3.3.1 Zielsetzung

Die Architektur *Residual Network* (ResNet) gehört zu den Klassifizierungsnetzwerken und wird, wie auch AlexNet, für die Einteilung eines gesamten Bildes in eine Klasse genutzt. Das Ziel war es, die in der Segmentierung lokalisierten Partikel einer Morphologie zuzuordnen. Die Korrektklassifizierungsrate der Bilder des Validierungsdatensatzes sollte wie bereits bei STEM-Bildern bei mindestens 75 % liegen. [69]

Für die Klassifizierung von SE-Partikeln wurde nach anfänglichen Tests ein komplexeres Netzwerk als AlexNet etabliert. Die Leistung von AlexNet für SE-Bilder war unzureichend. Tests ergaben gute Leistungen von ResNet34.

In der hier vorliegenden Arbeit wurde die Architektur unverändert von He et al. [19] übernommen und für die Einteilung von Nanopartikeln in REM-Bildern genutzt, welche im SE-Modus aufgenommen wurden. Die SE-Bilder sind pseudo 3-dimensionale Aufnahmen und zeigen eine komplexere Verteilung der Graustufen als STEM-Aufnahmen. Daher war es für diesen Bildtypus nötig, ein leistungsstärkeres neuronales Netz auszuwählen. ResNet ist lediglich der Oberbegriff für diesen Netzwerktyp. Für die hier vorliegende Arbeit wurde die 34 Schichten tiefe Variante ResNet34 genutzt.

3.3.2 Architektur

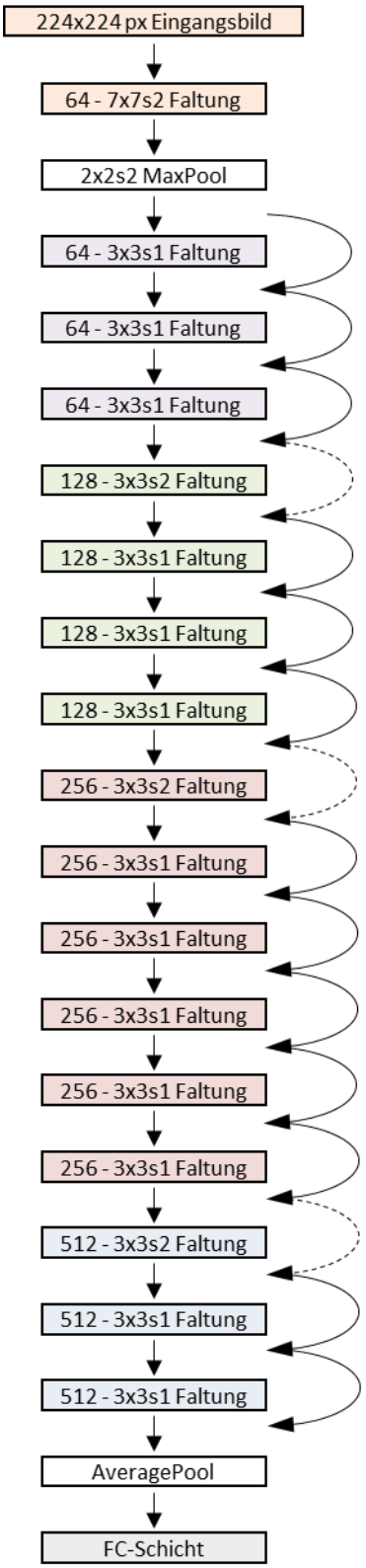


***Abbildung 21**: Aufbau von ResNet34. Das Eingangsbild besitzt eine Größe von 224x224 px. Zunächst findet eine 7x7 Faltung mit einer Schrittweite von $s = 2$ statt, welche die Eingangsgröße auf 112x112 px halbiert. Es werden 64 Filter in dieser Schicht verwendet (orange). Anschließend folgt eine MaxPool-Schicht mit einer Filtergröße von 2x2 und Stride 2, welche die Größe des Tensors erneut halbiert. Der Eigenschafts-Tensor besitzt nun die Größe 56x56x64 (x, y, z). Darauffolgend findet sich eine Reihe von Residual-Modulen wieder (lila). Innerhalb der Residual-Module durchläuft der Eigenschafts-Tensor 2 Faltungs-Schichten mit einer Filtergröße von 3x3 und $s = 1$. Die Anzahl der Filter pro Schicht beträgt auch hier 64. Nach 3 solcher Module erlebt der Feature Tensor ein Modul, welches mit selber Filtergröße eine Schrittweite von 2 aufweist. Die Anzahl der Filter wird auf 128 verdoppelt. Der resultierende Eigenschafts-Tensor besitzt nun eine Größe von 28x28x128 px. Die Abkürzungs-Verbindung dieses Moduls wird durch eine Repräsentations-Verbindung (gepunkteter Pfeil) ersetzt. Sie entspricht einer 1x1 Faltung mit $s = 1$. Es folgen 3 weitere Residual-Module (grün) mit 128 Filtern. Die Verdopplung der Filteranzahl bei $s = 2$ findet in 2 weiteren Faltungs-Schichten statt. Die Größe des Eigenschafts-Tensors reduziert sich dabei weiter von 16x16x256 px zu 8x8x512 px. Zuletzt findet ein* AveragePooling *statt, in welchem der Mittelwert jeder Eigenschafts-Karte, des Eigenschafts-Tensors in z-Richtung, gebildet wird. Der finale 512 Werte umfassende Vektor wird mittels einer vollvernetzten Schicht in eine Wahrscheinlichkeitsverteilung über alle Klassen umgewandelt.*

3.3.3 Datensatz

Der Datensatz für das Training des Netzwerkes wurde eigens für diese Arbeit erstellt. Zunächst wurden mehrere 100.000 einzelne Partikelbilder aus ca. 1000 vollständigen SE-Aufnahmen ausgeschnitten und auf 224x224 Pixel skaliert. Das Verhältnis von Höhe zu Breite wurde wie bereits bei STEM-Partikeln nicht verändert. Leere Bildbereiche wurden mit Nullwerten aufgefüllt (*Zero-Padding*). Die Partikel wurden aus Bildern diverser Synthesen entnommen und enthielten eine hohe Variabilität der Größe und elementarer Zusammensetzung. So wurden Au, Ag, SiO_2, ZnO, Calciumphosphat und TiO_2 Partikel verwendet, welche von vorherigen Mitgliedern des Arbeitskreises Epple synthetisiert worden waren.

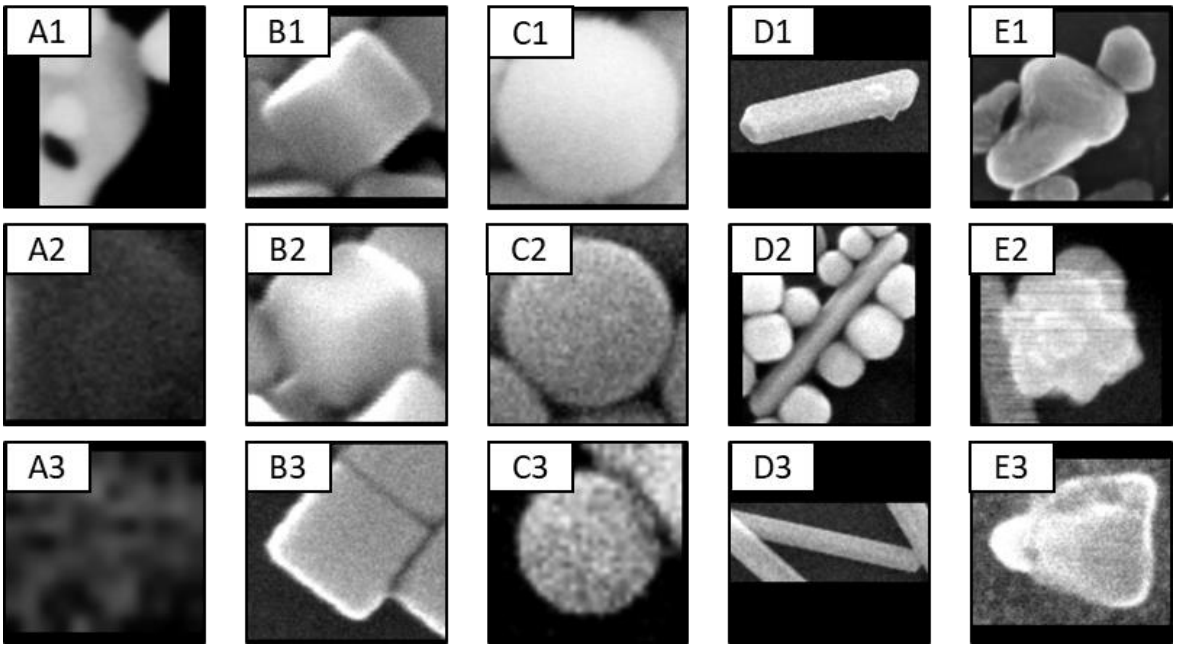


Abbildung 22: *Partikel des SE-Datensatzes, welche für das Training genutzt wurden. Die ursprüngliche reale Größe der Partikel ist für das Modell nicht von Belangen. Die Bildgröße beträgt 224x224 Pixel. Das Aspekt-Verhältnis wurde beim Ausschneiden nicht verändert, um die Morphologie nicht zu verzerren. Stattdessen wurden die Bilder mit Nullwerten aufgefüllt (*Zero-Padding*). Es sind Bilder folgender Morphologien im Datensatz enthalten: Hintergrund (**A**), Würfel (**B**), Kugel (**C**), Stab (**D**), Kugelartige/Verformte (**E**). Der schwarze Rand um die jeweiligen Bilder entspricht nicht dem Zero Padding. Lediglich große Bereiche wie in A1 und D1, sind mit Nullwerten aufgefüllte Bereiche.*

Im Anschluss wurden ca. 17.500 Bilder in fünf Klassen sortiert. Dies bedeutet, dass ca. 90 % der ausgeschnittenen Partikel verworfen wurden. Die verwendeten Klassenbezeichnungen

sowie die Anzahl der Objekte pro Klasse waren: Hintergrund & verdeckte Partikel, Würfel, Kugel, Stäbchen und Kugelartige & verformte Partikel (**Abbildung 22**). Bei der Auswahl der Partikel wurde auf eine hohe Variabilität des Aussehens und Orientierung geachtet. Darüber hinaus wurden ca. 1.400 Partikel in den Kategorien Hexagonal-Planar, Oktaeder, Pentagonal-Bipyramidal, Pyramide, Tetraeder und Trigonal-Planar gefunden. Diese erreichten allerdings nicht die für komplexe Objekte benötigten Bilder pro Klasse, sodass sie verworfen wurden. Für pseudo 3-dimensionale Bilder wie in SE-Aufnahmen werden 1.000-5.000 Objekte pro Klasse benötigt, um ein Modell sicher zu trainieren. [25]

Die so gewonnenen Bilder wurden im Verhältnis 80:20 in einen Trainingsdatensatz von ca. 14.000 und einen Validierungsdatensatz von ca. 3.500 Bildern aufgeteilt. Die Bildklassen waren in Trainingsdatensatz wie auch Validierungsdatensatz zu gleichen Teilen enthalten.

3.3.4 Training

Das Training erfolgt wie bereits bei STEM-Aufnahmen mittels Gradientenabstiegsverfahren. Das Netzwerk wurde nach He et al. initialisiert [49] und nutzt die Kreuzentropie aus Kostenfunktion (**Formel 2**). [43] Wie bereits im Training von AlexNet für STEM-Partikel, wurde auch der SE-Datensatz in einen Trainingsteil und einen Validierungsteil aufgeteilt.

Die Trainingsbilder wurden identisch zur Trainingsroutine von AlexNet, augmentiert, um eine Variation der Bilder zu erreichen. Die Data-Augmentation beinhaltete folgende Schritte:

i. Drehung um bis zu 360°
ii. Horizontales & vertikales Spiegeln des Bildes
iii. Bis zu 15 % Zoom
iv. Translation/Scherung der Bildkanten mit bis zu 15 % der Kantenlänge
v. Ausschneiden eines Bildbereiches mit 90 % der Kantenlänge des Originalbildes
vi. Addition von Gauß‘schem Rauschen mit $\sigma = 0$ und $\mu = 10$ für jedes Pixel
vii. Normalisierung auf das Intervall [-1, 1]

Nach der Augmentation werden je 32 Trainingsbilder zu einem Mini-Batch vereinigt. Dies ist eine geringere Anzahl an Bildern als in der Trainingsroutine von AlexNet, was mit dem höheren Rechenaufwand der Architektur begründet ist. Die Parameteroptimierung des Modells erfolgte nach dem Zeigen jedes Mini-Batches. Nachdem alle Mini-Batches des

Trainingsdatensatzes dem Modell gezeigt wurden, endete eine Epoche und das Model wurde validiert. Die Validierung erfolgte mit allen unveränderten Bildern des Validierungsdatensatzes. Im Anschluss wurden die Trainingsbilder neu augmentiert und eine neue Epoche begann. Die Reihenfolge, in welcher die Bilder pro Epoche dem Netzwerk gezeigt wurden, variierte ebenfalls. Das Training wurde beendet, sobald 20 aufeinanderfolgende Epochen keine Verbesserung in der Erkennung von Validierungsbildern mehr zeigten.

3.4 Erzeugung künstlicher SE-Bilder mittels Blender & CycleGAN

Während der Erstellung der Datensätze für das Training der bisher dargestellten Netzwerke ergab sich das Problem der Mehrdeutigkeit von Daten sowie des menschlichen Bias in der Auswertung von REM-Bildern. Die Mehrdeutigkeit ergibt sich bereits bei der Erstellung von Klassifizierungsmasken für das Training der Segmentierungs-Netzwerke. Für die manuelle Erstellung der binären Segmentierungen reproduzierte der Mensch das Bild in 2 Farben (Vordergrund und Hintergrund). Da REM-Bilder verrauscht und teilweise verschwommenen sind, führte die Erstellung von Annotationsmasken unweigerlich zu Fehlern. Das Rauschen in REM-Bildern begründet sich in der Streuung der Elektronen, wenn diese mit der Probe interagieren. Andere Einflüsse auf die Bildqualität sind Durchmesser des Probenstrahls, die Beschleunigungsspannung, Sputter-Material, Probenstruktur und Probenverteilung, um nur einige zu nennen. [65, 70] Diese Stellschrauben machen das „perfekte Bild“ nahezu unmöglich und erschweren es dem Menschen, eine genaue Unterteilung von Partikel und Probenträger in Vordergrund und Hintergrund vorzunehmen. Die Festlegung, wo der wahre Rand eines Partikels liegt, kann lediglich geschätzt werden. Bei mehreren Tausend Partikeln pro Bild werden dementsprechend bei der Erstellung der Klassifizierungsmaske mehrere Tausend Partikelgrenzen geschätzt. Dieses Problem wird deutlich größer, wenn die Erstellung eines Datensatzes für die Segmentierung mehrere Bilder mit jeweils einigen Hundert bis Tausend Nanopartikeln enthält.

Die Klassifizierung von Nanopartikeln in eine definierte Form von einer Gruppe von Menschen ist ebenfalls nicht eindeutig durchzuführen. Die durchschnittliche Sicherheit, mit der eine Gruppe von Menschen ein einzelnes Nanopartikel einer Morphologie zuordnet, beträgt ca. 75 %. [69] Das Problem ist die Mehrdeutigkeit der Partikelform aufgrund von Rauschen und

Verschwommenheit der Bilder. Durch diese Eigenschaften der Bilder ist die große Menge an Partikeln zu erklären, die im Zuge der Erstellung der Einzelpartikel-Datensätze zwar aus REM-Bildern ausgeschnitten, dann aber verworfen wurden. Von mehreren Hundert Partikeln pro REM-Bild waren oft nur wenige derart ausgeprägt in ihrer Form, dass sich diese für das Training einer KI eigneten.

Ein weiterer Punkt bei der Auswertung von REM-Bildern ergibt sich aus dem menschlichen Bias. [69, 71] Der menschliche Bias ist von der Erfahrung des Menschen als auch von Erwartungen und Wünschen geprägt. [72] Der Bias kann den menschlichen Analysten dazu verleiten, das zu sehen, was er erwartet, auch wenn die Synthese eigentlich fehlgeschlagen ist und die Partikel des Bildes in keiner Weise den gewünschten Ergebnissen entsprechen.

Bias und menschliche Unsicherheiten beeinflussen die Qualität der Datensätze, welche für Training und Validierung der Modelle genutzt werden. In dieser Arbeit wurde daher ein Ansatz verfolgt, den menschlichen Einfluss auf die Trainingsdaten zu minimieren. Künstliche Daten in Form von synthetischen REM-Bildern mit zuvor definierten Eigenschaften sollten den menschlichen Einfluss auf die Datensätze auf null reduzieren. Neuronale Netze, trainiert auf künstliche Datensätze, haben bereits in der Vergangenheit beachtliche Ergebnisse hervorbringen können. [21, 61, 67]

Künstliche Datensätze wurden in der Literatur bisher auf 3 unterschiedliche Weisen generiert: (i) Zum einen können die Bilder direkt mittels geeigneter Software gerendert werden. [21, 61] Die Software „Blender“ bietet hierfür eine geeignete Umgebung. [73] (ii) Eine andere Möglichkeit ist es, die Daten mittels Generativer Adversarialer Netzwerke (GAN) zu erzeugen. [47, 67] GANs sind ein Verbund aus mehreren neuronalen Netzen. GANs lernen aus realen Bildern neue Bilder zu generieren. Diese neuen Bilder werden in einem Stil synthetisiert, welcher dem GAN vorher beigebracht wurde. (iii) Die aufwendigste Methode besteht aus Monte-Carlo Simulationen, welche in dieser Arbeit nicht zur Anwendung kamen. [74, 75]

Blender und GANs können Bilder erzeugen, die echten REM-Bildern im SE-Modus sehr ähnlich sehen. Jedoch bestanden bisherige Probleme in den Details. So konnten Bilder, welche mittels Blender erstellt wurden, nicht die Art von Rauschen abbilden, welches in REM-Bildern präsent ist. Auch wurden die bisherigen synthetischen Bilder aus Blender-Simulationen optisch keinen REM-Aufnahmen nachempfunden, sondern Helium-Ionen-mikroskopischen (HIM) Aufnahmen. [61] Die Morphologie, Position und Größe von Partikeln kann innerhalb der

Software exakt eingestellt werden. Auch ist es möglich, neben dem gerenderten synthetischen REM-Bild und den Positionen aller Partikel der Szenerie, fehlerfreie synthetische Segmentierungen zu exportieren. Höhenkarten der Szenerie sind ebenfalls möglich. Die erzeugten Höhenkarten sollten dann durch ein GAN optisch dem REM angenähert werden.

Die Begründung, Höhenkarten zu nutzen, findet sich in einer Einflussgröße auf Qualität und Graustufenverteilung von REM-Bildern. Die relative Höhe von aufeinanderliegenden Partikeln hat einen direkten Einfluss auf die Intensität, mit der gestreute Elektronen den Detektor erreichen. Partikel, die exponierte Positionen aufweisen, zeigen eine durchschnittlich höhere Intensität dadurch, dass mehr gestreute Sekundärelektronen den Detektor erreichen. Partikel, welche von weiteren Partikeln teilweise verdeckt werden, streuen zwar in selben Maße Elektronen wie exponierte und frei liegende Partikel, jedoch können gestreute Sekundärelektronen erneut mit der Probe interagieren. Das Signal schwächt sich ab. **Abbildung 23** verdeutlicht dies.

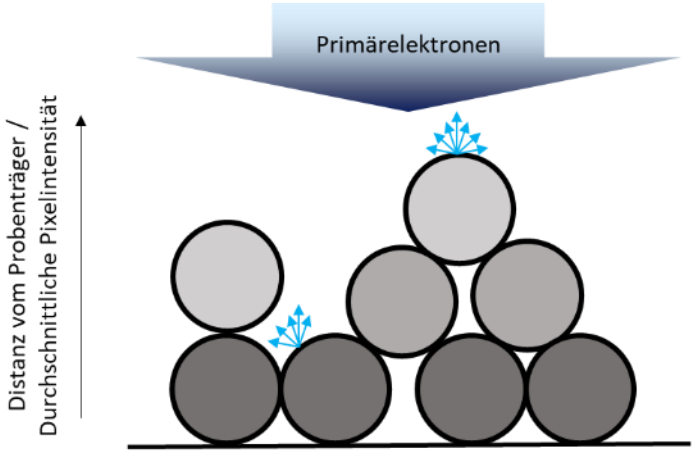


***Abbildung 23**: Vereinfachte Darstellung der mittleren Verteilung der Graustufen innerhalb eines REM-Bildes im SE-Modus in Abhängigkeit von der Probenhöhe. Die Graustufen des Bildes sind abhängig von der Anzahl der detektierten Elektronen an einem Punkt der Probe. Die detektierte Elektronenanzahl ist unter anderem abhängig von der Höhe und Überdeckung eines Partikels. Sekundärelektronen, welche von freiliegenden Partikeln stammen, erreichen häufiger den Detektor und liefern daher eine höhere Intensität (Graustufe) dieses Pixels. Es ist zu beachten, dass die Grafik nicht die typische Helligkeit von Kanten der Partikel (Fresnel-Effekt) berücksichtigt. Sie zeigt nur die mittlere Graustufenverteilung von aufeinanderliegenden Partikellagen. (Die Abbildung berücksichtigt keine unterschiedlichen Strahldurchmesser, Beschleunigungsspannungen, Probenströme oder Materialien.)*

GANs wiederum sind neuronale Netze, die bisher gute Ergebnisse in der Bildsynthese gezeigt haben. [47] Um die Informationen der zuvor simulierten Höhenkarte zu nutzen, wurde eine Weiterentwicklung des einfachen GAN genutzt, das CycleGAN. [48] CycleGAN gehört zur Klasse der bedingten GANs. Bedingte GANs nutzen ein Bild als Input, um ein realistisches Bild mit selbiger Semantik, aber definierter Erscheinung zu generieren. [76, 77] Aus echten REM-Bildern lernte das CycleGAN die typische REM-Erscheinung auf Höhenkarten abzubilden, ohne jedoch den Inhalt des Bildes zu verändern. Dies setzt voraus, dass Inhalt bzw. Bedeutung und Stil des Bildes miteinander korrelieren. [26] Beispielsweise sind ein Foto und ein Gemälde eines Menschen zwei unterschiedliche Stile mit selber Semantik, beide können über ihren Inhalt miteinander korreliert werden. Die synthetischen Bilder entsprechen dem Stil, der für ihr Training genutzt wurde. Bestand der Trainingsdatensatz des GANs lediglich aus Bildern von SiO_2 Kugeln, wird das GAN auch nur Bilder herstellen können, welche SiO_2 Kugeln zeigen. Dies zieht die Problematik nach sich, dass der Inhalt des synthetischen REM-Bildes nur bedingt vom Menschen kontrolliert werden kann. Die Problematik von GANs liegt in der Variabilität der erzeugbaren Daten, die wiederum von den Trainingsdaten abhängt. Sind die Eigenschaften innerhalb des Trainingsdatensatzes echter REM-Bilder zu hoch, kommt es zum Kollaps des Systems. Die erzeugten Bilder sind nicht realistisch oder es werden nur wenige unterschiedliche Bilder erzeugt. [47] Ist der Datensatz zu gleichförmig, können ebenfalls keine variablen künstlichen Bilder erzeugt werden.

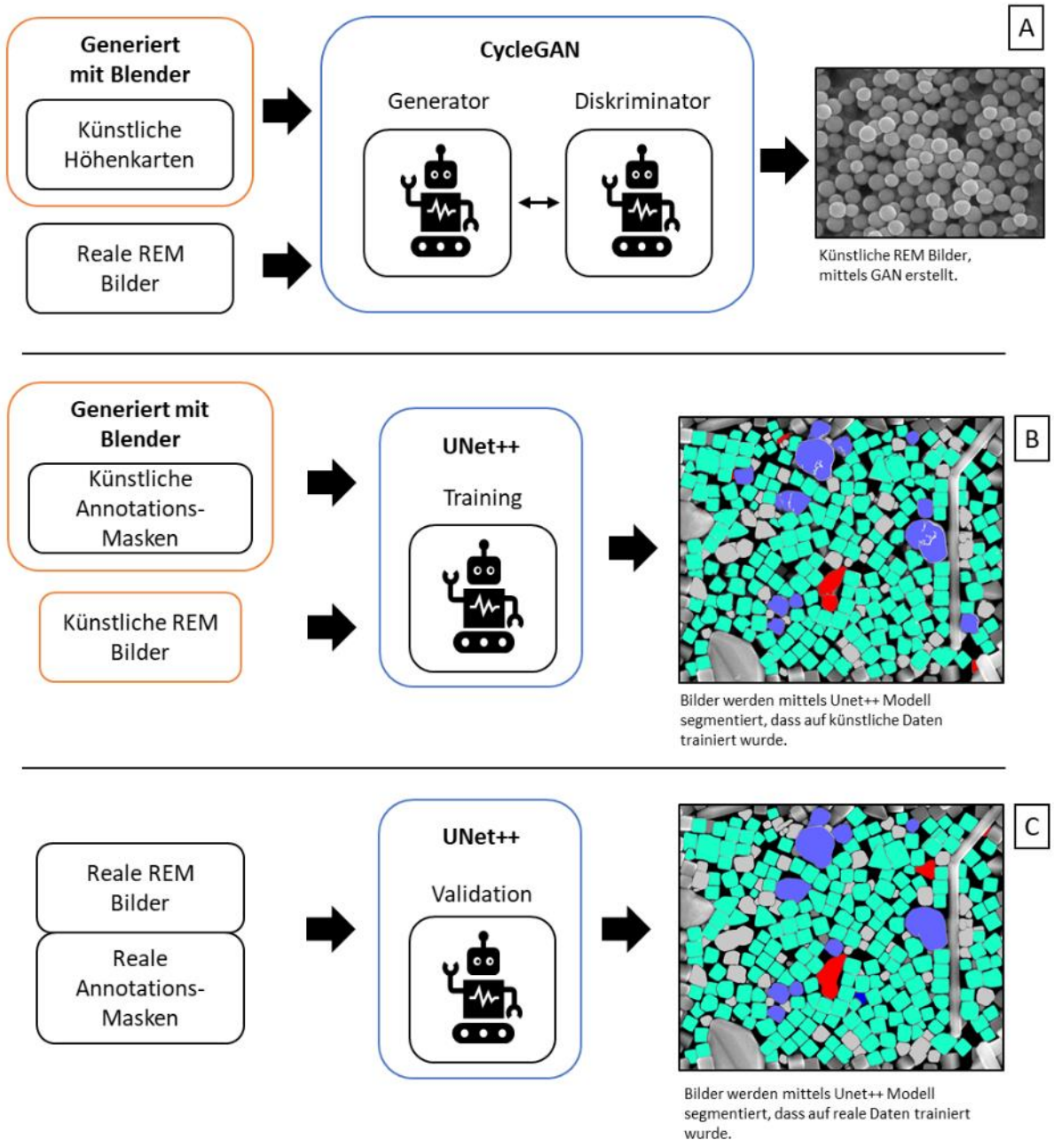


***Abbildung 24**: Die Abbildung demonstriert den Workflow für die Erstellung von künstlichen REM-Bildern aus simulierten Höhenkarten. Das CycleGAN wurde auf künstliche Höhenkarten und echte REM-Bilder trainiert. Das Training resultierte in der Synthese eines Datensatzes aus künstlichen REM-Bildern, deren Höheninformationen und Kontrast von den Höhenkarten abgeleitet sind (**A**). Um den Nutzen von künstlichen Daten zu überprüfen, wurde zunächst ein UNet++ Modell, zur Segmentierung von Bildern vollständig auf künstliche Daten trainiert (**B**). Ein identisches Netzwerk wurde auf reale Daten trainiert (**C**). Die Leistung beider Netzwerke wurde dann miteinander verglichen.*

Für die hier vorliegende Arbeit wurde eine Syntheseroute entwickelt, die die Vorteile beider bisherigen Systeme miteinander vereint. Zunächst wurden Partikel in Blender erschaffen und deren Fall auf einen Probenträger simuliert. Im Anschluss wurde von Blender je eine Höhenkarte und eine fehlerfreie Segmentierung exportiert. Die Höhenkarte diente neben echten REM-Bildern als Grundlage für das Training eines CycleGANs. Das GAN sollte nach dem Training synthetische REM-Bilder von Nanopartikel erzeugen, deren Positionen, Morphologie, Oberfläche und Intensitätsverteilung mit der Höhenkarte korrelierten. Die fehlerfreie Segmentierung und die synthetisierten REM-Bilder wurden dann in einem Datensatz für die Segmentierung zusammengefasst.

Durch die Verbindung von Blender-Simulationen und GAN sollte die Möglichkeit geschaffen werden, synthetische, fehlerfreie und fotorealistische REM-Bilder zu schaffen, um andere KI-Systeme darauf zu trainieren. Im Falle dieser Arbeit wurden die REM-Bilder im SE-Modus synthetisiert und anschließend für das Training eines UNet++ Segmentierung-Modells verwendet. Die Ergebnisse wurden mit dem UNet++ Modell verglichen, das mit menschlichen Daten trainiert wurde. **Abbildung 24** fasst diesen Prozess zusammen.

3.4.1 Blender

Blender ist eine Software zur Nachbearbeitung von Film oder Bildern. Dies schließt auch die Erstellung von vollständig künstlichen Umgebungen ein. Es ist in der Lage, einfache physikalische Simulationen durchzuführen und somit eine natürliche Verteilung von fallenden Objekten zu erreichen. Objekte werden vorher in 3D erstellt und anschließend über die eingebauten Mechanismen verteilt. Im Falle der Nanopartikel wurden verschiedene typische Formen erstellt. Ein Teil der so erstellten Formen ist in **Abbildung 25** dargestellt. Insgesamt wurden folgende Formen in unterschiedlichen Größen synthetisiert: Kugeln, Ikosaeder, Würfel und stumpfe Würfel, Oktaeder, Tetraeder, Pentagonale-Bipyramiden, trigonale und hexagonale Plättchen, hexagonale und runde Stäbchen sowie eine Variation von undefinierbaren Partikeln. Im Folgenden werden diese unförmigen Partikel als Kugelartige bezeichnet.

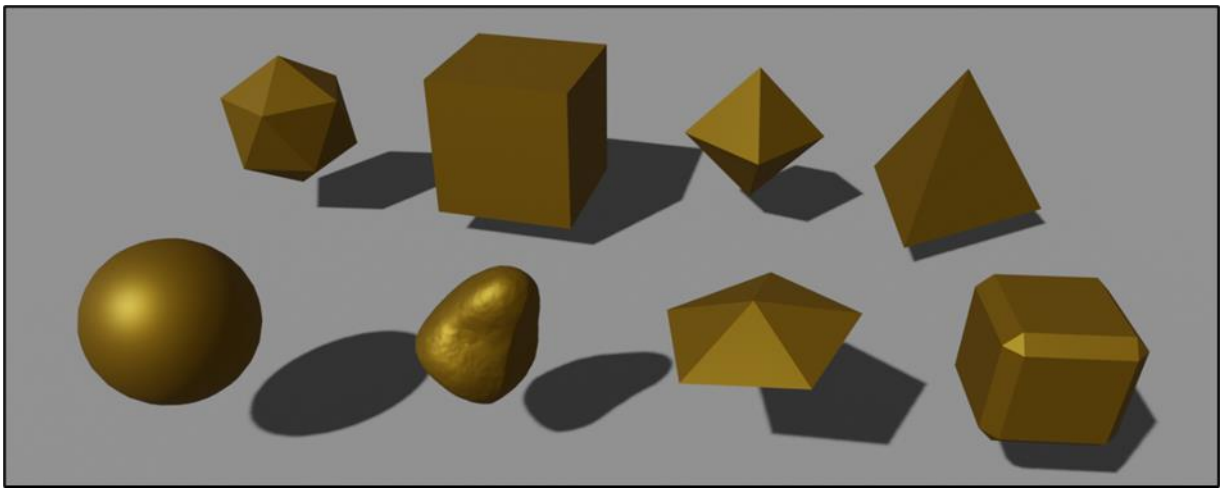


***Abbildung 25**: Typische Formen, die in Blender erstellt wurden. Licht, Schatten und Farbe dienen der anschaulichen Darstellung der Partikel und haben keinen Einfluss auf die erzeugten Daten.*

Für die Simulation wurden einige 100 bis 5000 dieser Partikel auf einen planaren Probenträger fallen gelassen. Jedes Partikel besaß in der Simulation physikalische Eigenschaften. Das heißt, es konnte von anderen Partikeln nicht durchstoßen werden. Partikel konnten sich beim Fallen gegenseitig abstoßen oder aufeinander liegen bleiben. Die Simulationen resultierten in realistischen Partikelverteilungen und Orientierungen, wie sie auch in echten REM-Bildern zu finden sind. Der Nachteil solcher Simulationen war die Dauer pro hergestelltem Bild. So konnten die Berechnungen von Partikelsystemen aus Tausenden Partikeln bis zu mehreren Stunden dauern.

Nachdem die Simulation abgeschlossen war, wurden 2 Bilder exportiert. Das erste Bild enthielt alle in der Szenerie sichtbaren Außenkanten der Partikel. Das zweite Bild war eine Höhenkarte, in der der Abstand jeder sichtbaren Fläche zum Probenträger dargestellt wurde. Aus den Außenkanten aller Partikel konnte dann eine Segmentierung berechnet werden. **Abbildung 26** zeigt dies.

Insgesamt wurden 27 2048x2048 Pixel große Bilder mit unterschiedlichen Partikelverteilungen simuliert. Die Bilder enthielten bis zu mehrere Tausend Partikel in den oben genannten Formen. Bei den Simulationen wurde darauf geachtet, dass Partikelformen, die in natürlichen Synthesen gemeinsam vorkommen, auch in den simulierten Bildern gemeinsam auftraten. Zum Beispiel zeigen Partikeldistributionen, die primär aus Würfeln bestanden, auch Tetraeder und Oktaeder.

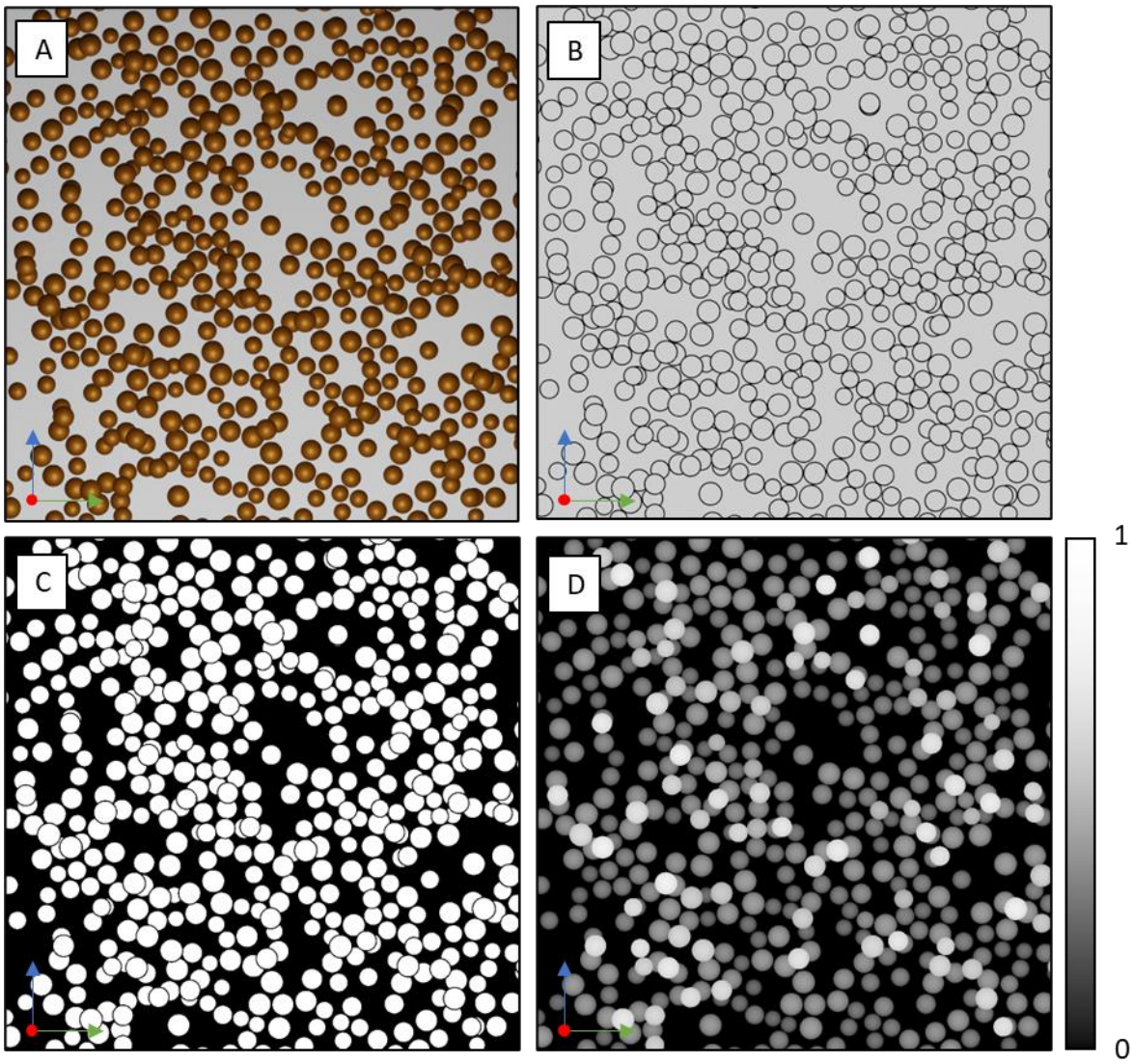


***Abbildung 26**: Typische mittels Blender generierte Bilder von Kugeln. Die Außengrenzen aller Partikel (**B**) und die Höhenkarte (**D**) konnten direkt aus Blender gewonnen werden. Daraus konnte die fehlerfreie Segmentierung (**C**) berechnet werden. Licht, Schatten und Farbe dienen der anschaulichen Darstellung der Partikel in (**A**) und haben keinen Einfluss auf die erzeugten Daten.*

3.4.2 REM-Bildsynthese mittels CycleGAN

Anders als Netzwerke zur Klassifizierung oder Segmentierung bestehen generative Netzwerke aus mehreren neuronalen Netzen, welche miteinander wechselwirken und voneinander lernen. [25] Generative Netzwerke fallen damit in die Klasse des unüberwachten Lernens. [39] Das GAN enthält mindestens zwei Bausteine: Den Generator, der Bilder aus Rauschen oder anderen Bildern synthetisiert und dem Diskriminator, welcher die Qualität der erzeugten Bilder überwacht. Der Diskriminator lernt, künstliche Bilder und reale Bilder zu unterscheiden. Daher benötigen generative Netzwerke weiterhin einen vom Menschen erstellten Datensatz. Die Parameter beider Netzwerke werden anhand der Bewertung des Diskriminators angepasst. Innerhalb des Trainings wird der Generator besser darin, künstliche Bilder zu erzeugen, während der Diskriminator besser darin wird, diese von realen Bildern zu unterscheiden. Mit zunehmender Trainingsdauer wird der Generator den Diskriminator übertrumpfen. Das Training endet, wenn der Diskriminator keine künstlichen Bilder mehr identifizieren kann.

3.4.3 Architektur

Das hier verwendete CycleGAN besteht aus 4 separaten CNNs. Zwei CNNs dienen der Erstellung von Bildern (Generatoren), während zwei weitere CNNs die Leistung der Generatoren überwachen (Diskriminatoren). Der Aufbau eines CycleGANs ist in **Abbildung 27** dargestellt. Die Architektur der Generatoren entspricht einem Autoencoder. Die Diskriminatoren entsprechen Klassifizierungsnetzwerken. Beide in diesem GAN verwendeten Netzwerktypen sind von Johnson et al. entworfen. [78] Der erste Generator lernt synthetische REM-Bilder aus simulierten Höhenkarten zu generieren. Der zweite Generator lernt synthetische Höhenkarten aus echten REM-Bildern zu generieren. Die Qualität beider Generatoren wird vom jeweiligen Diskriminator überwacht.

Um eine solche Netzwerk-Kombination mittels Gradientenabstiegsverfahren trainieren zu können, wird eine Kostenfunktion benötigt, die die Stärke der benötigten Parameteranpassung berechnet. Beim überwachten Lernen nutzt die Kostenfunktion die Vorhersage des Modells und das wahre Ergebnis für ein gegebenes Trainingsbeispiel, um die Kosten zu berechnen. Im Falle des CycleGAN existiert zu den simulierten Höhenkarten jedoch

kein echtes REM-Bild, welches in einer Kostenfunktion genutzt werden kann. Das CycleGAN nutzt daher drei Kostenfunktionen, wobei jede Kostenfunktion indirekt die Qualität des synthetisierten REM-Bildes misst.

Die erste Kostenfunktion setzt sich aus den Generatorkosten zusammen (*Generator-Loss*, L_G). Es gibt an, wie gut beide Generatoren in der Erstellung von real wirkenden Bildern sind. L_G setzt sich aus den individuellen Kosten der Generatoren zusammen. Generator I (G_1) nutzt eine simulierte Höhenkarte, um daraus ein synthetisches REM-Bild zu generieren. Dieses wird vom 1. Diskriminator (D_1) bewertet. Für die Bewertung von synthetischen REM-Bildern lernt D_1 synthetische und reale Bilder zu unterscheiden. Generator II (G_2) nutzt reale REM-Bilder, um daraus eine synthetische Höhenkarte zu generieren. Dieses wird vom 2. Diskriminator (D_2) bewertet. Für die Bewertung von synthetischen Höhenkarten lernt D_2 synthetische und simulierte Höhenkarten zu unterscheiden. Je schlechter die Diskriminatoren in ihrer Unterscheidung abschneiden, desto höher fallen die Generatorkosten aus. Die **Formeln 21** bis **23** fassen dies zusammen. $x_i \in X$ ist eine Höhenkarte des Datensatzes. $y_k \in Y$ ist ein reales REM-Bild.

$$L_{G_1} = D_1(y_k, G_1(x_i)) \quad 21$$

$$L_{G_2} = D_2(x_i, G_2(y_k)) \quad 22$$

$$L_G = L_{G_1} + L_{G_2} \quad 23$$

Die zweite Kostenfunktion bewertet, ob die Partikelpositionen in der simulierten Höhenkarte den Partikelpositionen in dem synthetischen REM-Bild entsprechen (*Cycle Consistency-Loss*, L_C). Eine simulierte Höhenkarte wird mittels Generator 1 in ein synthetisches REM-Bild überführt. Das synthetische REM-Bild wird von Generator 2 zurück in eine Höhenkarte überführt. Wenn beide Generatoren perfekt arbeiten, sollte die zyklische Höhenkarte der ursprünglichen Höhenkarte entsprechen. Die Differenz beider Bilder wird über den mittleren quadratischen Abstand der einzelnen Pixel berechnet. Der Hintergrund liegt in folgender Annahme: Wenn einer der Generatoren die Position der Partikel verändert, wird die rückgeführte Höhenkarte eine andere Pixelverteilung aufweisen als die ursprüngliche Höhenkarte. Da dies ein fehlerhaftes Verhalten der Netzwerke ist, kann eine Fehlerfunktion eingesetzt werden um den mittleren Abstand der richtigen (simulierten) und vorhergesagten

(zyklischen) Höhenkarte zu messen. Der mittlere quadratische Abstand über alle Pixel (n) des Bildes entspricht dann den Kosten. Die genutzte Fehlerfunktion ist in **Formel 24** gegeben:

$$L_c = \frac{1}{n}\sum\left(G_2\left(G_1(x_i)\right) - x_i\right)^2 \qquad 24$$

Die dritte Kostenfunktion beschreibt die Identitätskosten (*Identity-Loss*, L_i). Die Identitätskosten gründen in folgender Annahme: Da der erste Generator reale REM-Bilder schaffen soll, müsste dieser, wenn ein reales REM-Bild als Input genutzt wird, das Bild unverändert lassen. Jede Manipulation eines echten REM-Bildes seitens des Generators würde die Echtheit des Bildes verringert. Die Identitätskosten fasst diese Annahme in folgender Kostenfunktion zusammen (**Formel 25**).

$$L_i = \frac{1}{n}\sum(G_1(y_k) - y_k)^2 \qquad 25$$

Alle Kostenfunktionen werden für das Training genutzt. Experimente haben gezeigt, dass eine individuelle Gewichtung der einzelnen Kostenfunktionen zu realistischeren Bildern führen. [48, 67] Die vollständigen Kosten pro Trainingsbild ergeben sich nach **Formel 26**.

$$L = 10 * (L_{G_1} + L_{G_2}) + L_c + \frac{L_i}{8} \qquad 26$$

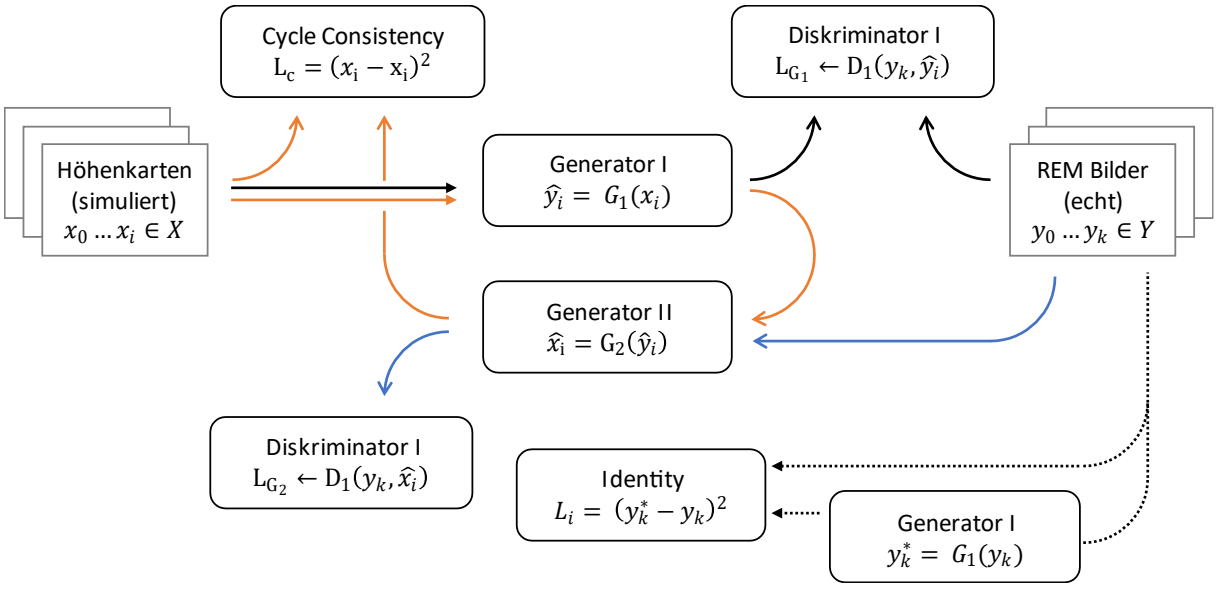


***Abbildung 27**: Aufbau eines CycleGAN nach Zhu et al. [48] Das GAN nutzte einen Datensatz aus simulierten Höhenkarten X und echten REM-Bildern Y, um die Transformation von Höhenkarten in fotorealistische REM-Bilder zu erlernen. Da die Bilder beider Datensätze unterschiedliche Ausschnitte möglicher Partikelverteilungen zeigen, gibt es keine passenden Paare von einzelnen Bildern. Es kann daher keine klassische Kostenfunktion wie im überwachten Lernen genutzt werden. Stattdessen werden für das Training des Generators G_1 und des Hilfsgenerator G_2, zwei bewertende Netzwerke D_1 , D_2 verwendet, um den angestrebten Realismus zu verwirklichen. Diese Netzwerke wurden genutzt, um unterschiedliche Kosten zu berechnen. Die Beziehung zwischen den einzelnen Netzwerken, Datensätzen und Kostenfunktionen sind mit farbigen Pfeilen markiert: Generatorkosten sind in schwarz (L_{G_1}) und blau (L_{G_2}), Zyklischen Kosten in orange (L_c) und die Identitätskosten schwarz gestrichelt (L_i) dargestellt. Der Generator I ist zur besseren Sichtbarkeit zwei Mal abgebildet. Es handelt sich allerdings um dasselbe Netzwerk.*

3.4.4 Datensatz

Der für das Training des GANs genutzte Datensatz umfasste 27 simulierte Höhenkarten und 30 echte REM-Bilder. Es handelte sich bei den REM-Bildern um dieselben Bilder, die bereits im Training des UNet++ Modells Verwendung fanden. Aus den Bildern wurden insgesamt 700 Patches zu je 512x512 Pixel Größe ausgeschnitten. Aus diesen Patches wurden zufällig 140 Patches zur Validierung des Modells ausgewählt. Es handelte sich um einen statischen Datensatz, die Patches wurden während des Trainings nicht neu generiert. Experimente mit verschiedenen Datensätzen haben gezeigt, dass ein statischer Datensatz zu einem stabileren Training und somit zu realistischeren Bildern führt.

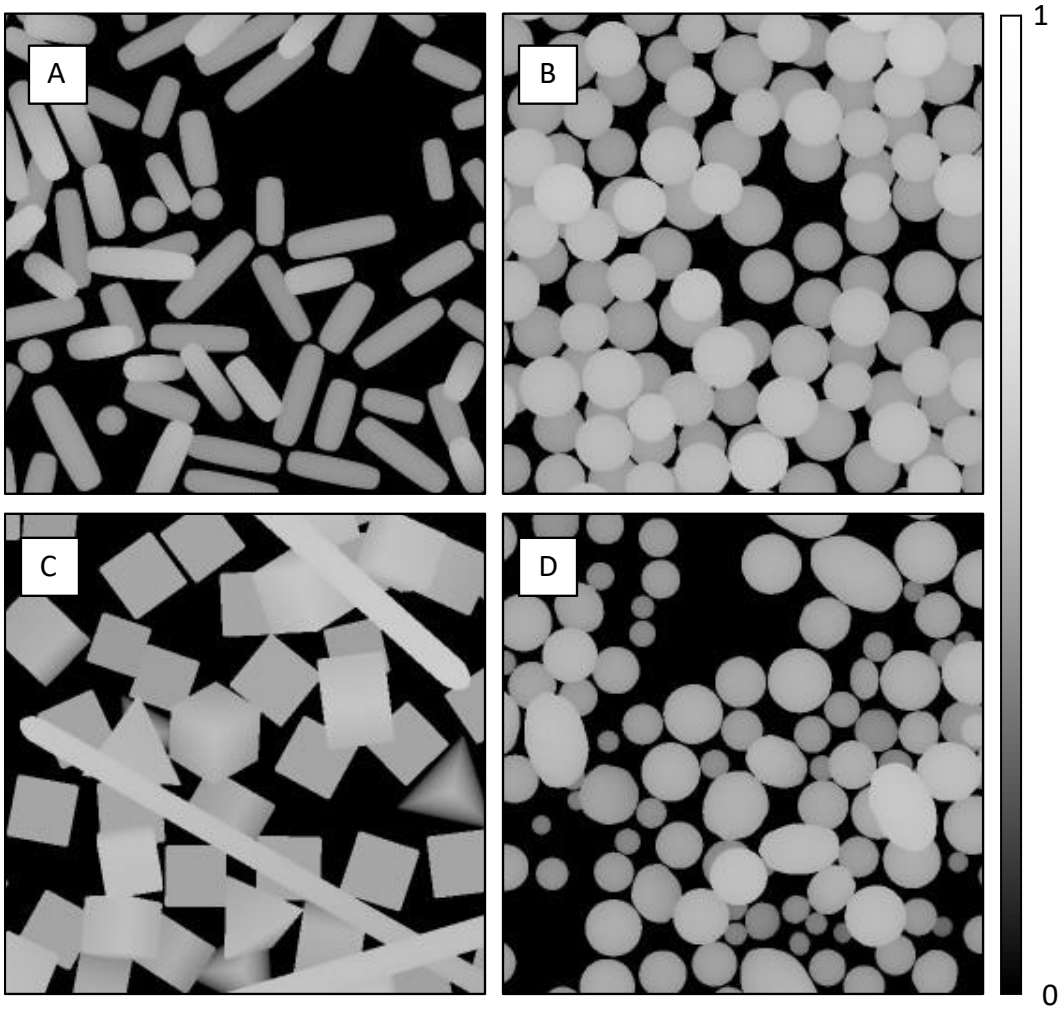


***Abbildung 28**: Typische Patches generiert aus Höhenkarten. Die gezeigten Partikelformen sind typische Formen von Nanopartikeln, wie sie auch in dem Datensatz der realen REM-Bilder vorhanden sind. Kurze Stäbchen und Kugeln, wie sie in SiO_2 Proben vorhanden sind (**A**, **B**).*

*Würfel, Stäbchen und Tetraeder ähnlich zu Ag-Nanopartikeln (**C**) und gemischte Partikelformen und Größen, die in mehreren Synthesen auftreten können (**D**).*

3.4.5 Training

Bei einem GAN handelt es sich um einen Zusammenschluss mehrerer CNNs. Daher erfährt jedes Netzwerk eine eigenständige Parameteranpassung. Jedes Netzwerk wird eigenständig mittels des Gradientenabstiegsverfahrens trainiert. Die Gesamtkosten aller Netzwerke sind in **Formel 33** gegeben. Die Kosten aller Netzwerke bilden die Grundlage der Anpassung der Parameter mittels des Optimizers. Als Optimizer erhält jedes Netzwerk eine eigene Version von ADAM. Der ADAM-Algorithmus ist in der Lage, adaptiv in den Trainingsprozess einzugreifen. Dadurch erfahren die Netzwerke minimal unterschiedlich starke Parameteranpassungen. [46]

Die Kostenfunktion ist definiert als die Abweichung (Fehler) des vom Modell geschätzten Ergebnisses vom wahren Ergebnis. [25] Im überwachten Lernen nimmt durch den Trainingsprozess der Fehler mit zunehmender Trainingsdauer ab. Das Training endet, sobald der Fehler, gemessen mittels der Validierungsdaten, nicht weiter abnimmt. Im Falle des CycleGAN war der Gesamtfehler von 3 Kostenfunktionen abhängig. Das Training des CycleGANs war somit definiert durch 3 gewünschte Eigenschaften, die das CycleGAN im Training erlernen sollte. Dies bedeutete, dass die Fehler der individuellen Kostenfunktionen sehr stark schwankten. Eine Anpassung des Netzwerkes bezüglich der einen Eigenschaft minimierte die Kostenfunktion, welche diese Eigenschaft misst. Beispielsweise die Partikelposition gemessen mittels zyklischer Kostenfunktion. Die Anpassung des Netzwerkes in dieser Richtung bedeutete aber nicht, dass die anderen Kostenfunktionen von derselben Parameteranpassung profitieren. Die Gesamtkosten sanken im Laufe des Trainings zwar, aber nicht genug, um aber eine minimale Anpassung der Parameter zu erreichen. Um diesem Effekt entgegenzuwirken und einen stetigen Abfall der Kosten zu erreichen, wurden diese künstlich verringert. Die modifizierten Kosten wurden dann als Grundlage des Trainings genutzt. Zwischen den Epochen 81-120 fielen die modifizierten Kosten linear ab. **Formel 27** verdeutlicht diesen Zusammenhang.

$$L_{mod} = \begin{cases} L \; wenn \; E \leq 80 \\ L * \left(1 - \frac{(E - 80)}{120}\right) \; wenn \; E > 80 \end{cases} \quad 27$$

Wobei L die berechneten Gesamtkosten, L_{mod} die modifizierten Kosten und E die Anzahl der bisherigen Epochen sind.

Die Netzwerke wurden für insgesamt 200 Epochen trainiert. Im Anschluss wurde Generator G_1 genutzt, um einen Datensatz von künstlichen REM-Bildern aus Höhenkarten zu erzeugen. Die künstlichen REM-Bilder wurden genutzt, um ein Segmentierungsnetzwerk zu trainieren. Auch wurde eine Umfrage unter chemisch erfahrenen Personen durchgeführt, um den Realismus der Bilder zu validieren.

Nachdem das GAN trainiert wurde, ist eine Umfrage unter den Mitgliedern des AK Epple durgeführt worden, um eine menschliche Beurteilung der Bilder, hinsichtlich ihres Realismus zu erhalten. Hierfür wurden den Befragten eine Reihe von echten und künstlichen REM-Bildern gezeigt. Die Aufgabe lag in der korrekten Einschätzung der Bilder.

3.5 UNet++ trainiert auf künstlichen Daten

3.5.1 Zielsetzung

Künstliche Daten in Form von Bildern und Annotationsmasken wurden für das Training eines UNet++ Modells genutzt. Ziel des Trainings war es, ein Modell zu erhalten, dessen Segmentierungsleistung mit einem auf menschliche Daten trainierten Modells verglichen werden konnte. Für den Vergleich der Systeme wurde der SE-Validierungsdatensatz des menschlichen UNet++ Modells verwendet. Da dieser Datensatz vom Menschen erstellt wurde, konnte gemessen werden, inwiefern sich die Leistung des künstlichen Modells vom menschlichen Modell unterschied.

3.5.2 Datensatz

Der verwendete Trainingsdatensatz bestand aus 27 Bildern. Es handelte sich um diese selben 27 Bilder, aus denen bereits der Datensatz für das Training des GANs erstellt wurde. Aufgrund der Dauer und Komplexität der Herstellung der Höhenkarten wurde auf die Generierung weiterer Höhenkarten verzichtet. Die Höhenkarten wurden mittels des Generators in simulierte REM-Bilder transformiert. Zu jedem Bild existierte eine fehlerfreie Segmentierung. Aus den Segmentierungen wurden distanzbasierte Gewichtungskarten berechnet, wie bereits für den SE-Datensatz im UNet++ Training.

3.5.3 Architektur & Training

Die Architektur entsprach der UNet++ Architektur, die bereits für SE-Bilder verwendet wurde. Das Netzwerk wurde identisch initialisiert und trainiert, wie das bereits verwendete UNet++ Modell.

4 Ergebnisse & Diskussion

Im Folgenden sollen die Ergebnisse der Bildsegmentierung, Partikelklassifizierung und der Synthese von künstlichen REM-Bildern gezeigt werden. Hierfür werden zunächst Ergebnisse der gesamten Validierungsdatensätze aufgezeigt. Im Falle der Bild-Segmentierungen wurde die Segmentierungsleistung der Netzwerke mittels gängiger Metriken quantifiziert. Zudem kann die mittlere Größe der im Datensatz annotierten Partikel für einen ersten Vergleich mit der KI genutzt werden. Die mittlere Größe der annotierten Partikel wurde mit den Partikelgrößen verglichen, die aus den Segmentierungen der KI berechnet wurden. Hierfür wurden die Partikelgrößen des gesamten Datensatzes genutzt. Aufgrund der Komplexität von SE-Bildern verglichen mit STEM-Bildern wurde eine Analyse bezüglich einiger Einflussfaktoren auf die Segmentierung vorgenommen.

Die Klassifizierung von Partikeln wurde ebenfalls mittels gängiger Metriken ausgewertet. Es wurde zudem eine Analyse des Netzwerks zur Klassifizierung von STEM-Partikel hinsichtlich der „Aufmerksamkeit“ des Netzwerkes vorgenommen. Dies sollte zeigen, worauf das Netzwerk achtet, wenn es Partikel klassifiziert.

Da die Datensätze nur einen Ausschnitt der Realität zeigen, werden im Anschluss ausgewählte Einzelbeispiele von STEM- und SE-Bildern analysiert.

Die Auswertung der Ergebnisse des GANs erfolgte in 2 Schritten. Schritt 1 war die qualitative Auswertung Bilder mittels einer Umfrage. Schritt 2 war die Auswertung des SE-Validierungsdatensatzes und weiterhin die Segmentierung von Einzelbildern mittels des auf künstlichen Daten trainierten Modells.

4.1 Bildsegmentierung mittels UNet++

Das Training einer KI zur Segmentierung von REM-Bildern wurde mittels dreier Datensätze durchgeführt: (i) Ein SE-Datensatz enthielt echte REM-Bilder und die zugehörigen binären Annotationen, die vom Menschen erstellt wurden. Zusätzlich enthielt der Datensatz für jedes Bild die distanzbasierten Gewichtungskarten. (ii) Einen STEM-Datensatz welcher ebenfalls echte REM-Bilder und die dazugehörigen binären, vom Menschen erstellten Annotationen enthielt. Dieser Datensatz enthielt intensitätsbasierte Gewichtungskarten. (iii) Ein SE-Datensatz aus künstlichen, mittels GAN erzeugten SE-Bildern und den künstlichen fehlerfreien Segmentierungen. Dieser nutzte erneut distanzbasierte Gewichtungskarten.

		Wahres Ergebnis	
		Positiv	Negativ
Vorhersage des Modells	Positiv	Echt Positiv True Positive (TP)	Falsch Positiv *False Positive* *(FP)*
	Negativ	Falsch Negativ *False Negative* *(FN)*	Echt Negativ *True Negative* *(TN)*

***Abbildung 29**: Herleitung der Einteilung von Pixeln in eine der vier möglichen Klassen. Aus den möglichen Kombinationen der vier Zustände ergeben sich die Metriken, mit denen die Leistung eines neuronalen Netzes gemessen wird.*

Alle Datensätze wurden in Trainings und Validierungs-Datensatz gespalten. Die Ergebnisse aller Bilder des Validierungsdatensatzes werden hier gemittelt zusammengefasst. Für die Quantifizierung der Leistung wurde die Größe der Bilder des Validierungsdatensatzes berücksichtigt. Die Leistung der Netzwerke wurde mit verschiedenen Metriken quantifiziert. Jede Metrik fokussiert dabei eine andere Messgröße, welche Aufschluss über die Probleme des jeweiligen KI-Systems gibt. Diese Messgrößen leiten sich aus den folgenden vier Möglichkeiten ab, die ein Pixel einnehmen kann: TP – *true positive*, FP – *false positive*, TN – *true negative*, FN – *false negative.*

Die Hauptmetrik ist die *Intersection-over-Union* (IoU). Diese misst den Anteil der Pixel, die in der Segmentierung und der Vorhersage des Modells übereinstimmen. Das Minimum sind 0 %, das Maximum ist sind 100 %. Je höher, desto besser. Eine IoU von mindestens 70 % wird als sehr gut für ein Segmentierungsmodell angesehen. [79]

$$Intersection\ over\ Union\ (IoU) = 100 * \frac{TP}{TP + FP + FN}\ [\,\%] \qquad 28$$

Die Nebenmetrik *Precision* ist ein Maß für den Anteil tatsächlich positiver Pixel, die vom Modell korrekt identifiziert wurden. Sie ist definiert als die Anzahl der wahren Positiven Pixel (d. h. Pixel, die korrekt identifiziert wurden), dividiert durch die Gesamtzahl der gemessenen positiven Fälle. Dies wird für Vordergrundpixel und Hintergrundpixel getrennt bestimmt.

$$Precision = 100 * \frac{TP}{TP + FP}\ [\,\%] \qquad 29$$

Die Nebenmetrik *Recall* ist ein Maß für den Anteil tatsächlich positiver Pixel, die vom Modell korrekt identifiziert wurden. Sie ist definiert als die Anzahl der wahren Positiven Pixel, dividiert durch die Gesamtzahl der tatsächlichen positiven Fälle.

$$Recall = 100 * \frac{TP}{TP + FN}\ [\,\%] \qquad 30$$

Das Minimum ist 0 %, das Maximum ist beträgt 100 %. Je höher die Metriken liegen, desto erfolgreicher ist die Segmentierung eines Bildes. Es ist wichtig zu beachten, dass Recall und Precision oft umgekehrt proportional zueinander liegen, was bedeutet, dass die Verbesserung des einen oft zu einer Verringerung des anderen führen kann. Daher ist es oft notwendig, zwischen Recall und Präzision abzuwägen. Grundsätzlich können aber beide Metriken ähnlich hoch liegen.

Zusätzlich zu diesen Metriken wurde der Rand Fehler bestimmt. Dieser basiert auf dem Rand Index. [80] Der Rand Index misst den Grad der Übereinstimmung der Segmentierung und der Vorhersage des Modells und berechnet, ob einzelne Partikel in beiden Bildern überlappen, das heißt an derselben Position liegen. Der Rand Index ist ein Maß für die Fähigkeit der KI, einzelne Partikel zu voneinander zu trennen. Dabei ist der Rand Index unbeeinflusst von der Partikelgröße oder der Korrektheit der Segmentierung. Er misst einzig, ob dort, wo ein Partikel in der wahren Segmentierung liegt, auch ein Partikel in der Vorhersage liegt, welches mit keinem weiteren wahren Partikel überlappt.

4.1.1 UNet++ trainiert mittels realer STEM-Aufnahmen

Die Leistung des Modells zur Segmentierung von STEM-Aufnahmen ist in **Tabelle 1** dargestellt. Für die Anwendung von UNet++ wurden die Gewichtungskarten an die starken Kontrastunterschiede in STEM-Bildern, die typischerweise zwischen Hintergrund und Partikel auftreten, angepasst. Um die Verbesserung durch die Anpassung der Gewichtungskarten zu testen, wurden zwei Netzwerke trainiert. Das erste Modell nutzte distanzbasierte Gewichtungskarten, während das zweite Modell intensitätsbasierte Gewichtungskarten nutzte. Beide Modelle wurden mit dem gleichen Validierungsdatensatz getestet (**Tabelle 1**).

***Tabelle 1**: Leistung der Modelle zur Segmentierung von STEM-Bildern. Ersichtlich ist die Steigerung der erreichten Segmentierung durch den Einsatz von intensitätsbasierten Gewichtungskarten.* [81]

	IoU	Precision	Recall	Rand Index
UNet++ *Distanzbasierte Gewichtungskarten*	88 ± 1 %	96 ± 4 %	91 ± 7 %	1,1 ± 0,6 %
UNet++ *Intensitätsbasierte Gewichtungskarten*	92 ± 7 %	96 ± 6 %	96 ± 4 %	1 ± 1

Beide Modelle performten ähnlich gut mit einer IoU über 70 %. Die Segmentierung ist in beiden Fällen sehr gut. Auch liegen Precision und Recall ähnlich hoch. Die geringe Abweichung der beiden Metriken zueinander zeigt, dass Vordergrund und Hintergrund mit ähnlicher Qualität segmentiert werden. Der Rand Fehler spricht mit etwa 1 % von einer sehr guten Trennung einzelner Nanopartikel.

UNet++ welches mittels intensitätsbasierter Gewichtungskarten trainiert wurde, zeigt fast durchweg eine höhere Leistung als die auf distanzbasierten Gewichtungskarten trainierte UNet++ Version. Die IoU stieg durch die Einbindung der Bildintensität um 4 Prozentpunkte gegenüber dem distanzbasierten Modell, welches die Leistung nicht berücksichtigt. Die Precision zeigt eine Differenz von 0,02 Prozentpunkten, welche zu vernachlässigen ist. Der Recall hingegen stieg um 4 Prozentpunkte. Der Rand Fehler konnte minimal verringert werden.

Die Einbindung der Bildintensität von kontrastreichen STEM-Bildern in die Gewichtungskarten zeigt einen deutlichen Vorteil gegenüber der reinen distanzbasierten Gewichtungsmethode,

wie sie Ronneberger et al. einführten. [56] Die Steigerung des Recalls zeigt, dass die Verbesserung der Gesamtsegmentierung, gemessen mittels IoU, von einer besseren Identifizierung von Hintergrundpixeln rührt. Die Steigerung der Metriken zeigt, dass Gewichtungskarten während der Segmentierung dem Netzwerk als Anleitung dienen, auf welche Pixel sich das Netzwerk „konzentrieren“ soll. Die Anleitung in Form von distanzbasierten Gewichtungskarten, welche lediglich die Distanz eines Pixels zu den nächsten zwei Partikeln berücksichtigen, reichte nicht aus, um Hintergrundbereiche zu segmentieren, die eine hohe Intensität besitzen. Diese hohe Intensität mag mehrere Ursachen haben. Zum einen können Partikel, die nahe beieinander liegen, nicht getrennt werden, da der Hintergrund zwischen den Partikeln eine hohe Intensität aufweist. Zum anderen können Bereiche des Bildes eine höhere mittlere Intensität aufweisen, sodass die Intensitätsänderung zwischen Partikel und Hintergrund (Probenträger) geringer ist oder nur leicht ansteigt. **Abbildung 30** veranschaulicht dies.

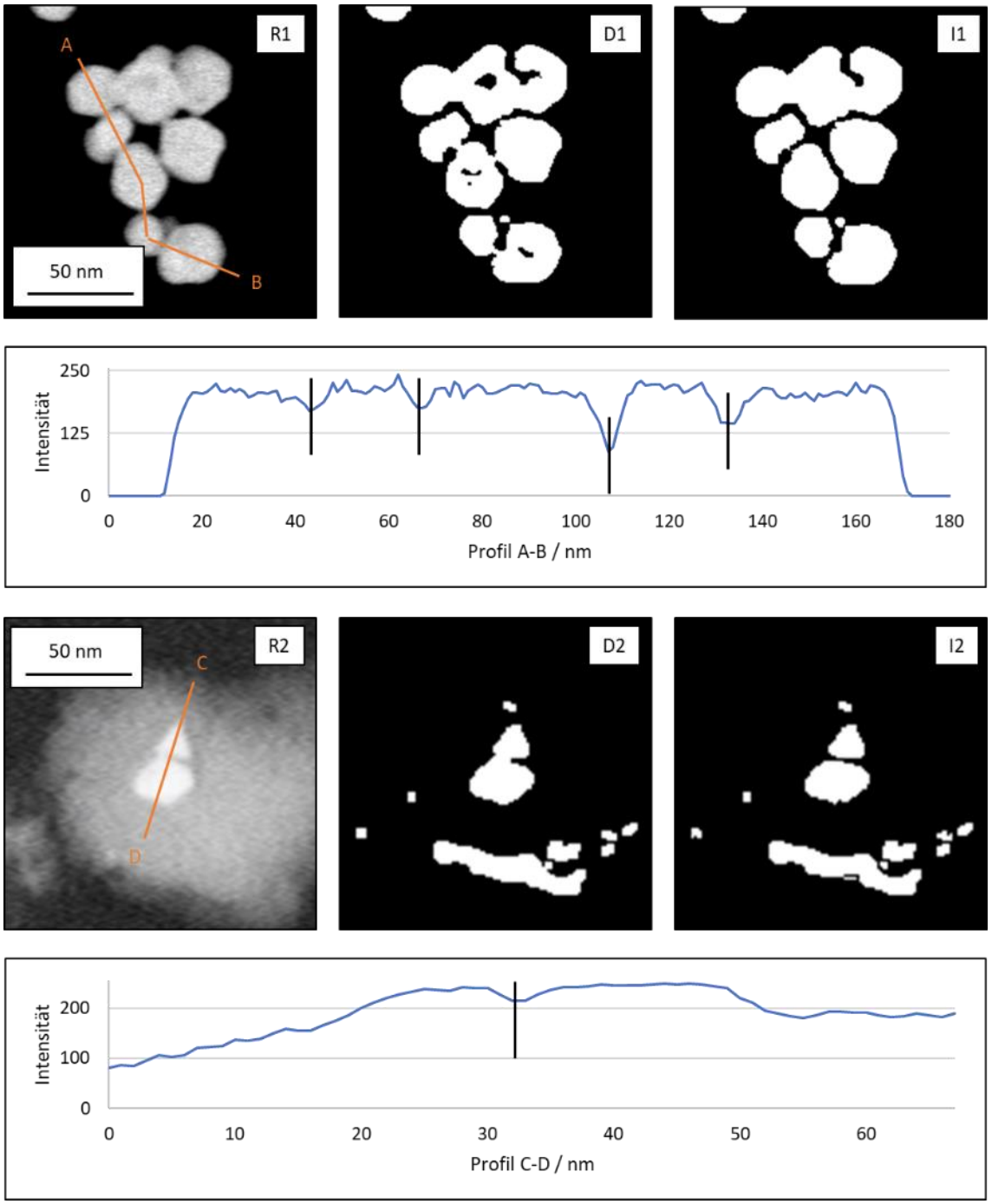


***Abbildung 30**: Die Abbildung zeigt zwei Fälle, in denen intensitätsbasierte Gewichtungskarten zu einer besseren Segmentierung führten. Beide Profile zeigen den geringen Intensitätsunterschied zwischen den Partikeln und ihrem umgebenden Probenträger. Schwarze Linien in den gezeigten Profilen heben den Bereich hervor, in dem der Mensch eine Trennung von zwei Partikel vornehmen würde. Das Profil A-B durchschneidet mehrere Partikel und den dazwischen liegenden Hintergrund. Hintergrundpixel, die zwischen zwei sich berührenden Partikeln liegen, weisen eine höhere Intensität auf als durchschnittliche Hintergrundpixel (schwarze Bereiche). Die hellen Hintergrundpixel überbrücken die Partikel*

*und formen ein kontinuierliches Aggregat (**R1**). Das Modell trainiert auf distanzbasierte Gewichtungskarten, berechnet eine Segmentierung, in der die einzelnen Partikel größtenteils zusammenhängen (**D1**), wohingegen das Model trainiert auf intensitätsbasierten Gewichtungskarten in der Lage ist, die meisten Partikel voneinander zu trennen (**I1**). Das Profil zeigt deutlich den Bereich, welcher von beiden Modellen getrennt werden kann. In diesem Bereich fällt die Bildintensität um ca. 70 Helligkeitsstufen ab. Das Profil C-D durchschneidet zwei Partikel in einem hellen Bereich des Bildes (**R2**). Erneut ist die Leistung des auf intensitätsbasierten Gewichtungskarten trainierten Modells besser (**I2**), als ohne Berücksichtigung der Intensität (**D2**). In der Trennung der Partikel. Es werden allerdings auch Bereiche als Vordergrund erkannt, die offenkundig keine Partikel sind. Die geschieht bei beiden Modellen.*

Mittels distanzbasierter Gewichtungskarten konnte das Modell nicht so weit trainiert werden, dass es Partikel zu trennt, die sehr nahe beieinander liegen. Mit der Einführung von intensitätsbasierten Gewichtungskarten wurde es möglich, solche Partikel voneinander zu trennen. Dies wird durch die Senkung des Rand-Fehlers deutlich. Eine höhere Anzahl von Partikeln, die pro Bild erkannt werden, macht es möglich, eine genaue statistische Aussage zur Größenverteilung von Nanopartikeln in STEM-Bildern zu treffen.

Der Validierungsdatensatz enthielt einige Bilder mit übereinanderliegenden Partikeln, sogenannten Agglomeraten. Diese können durch die vorgestellten Gewichtungskarten nicht adressiert werden. Die Überlagerung von Partikeln führt zu untrennbaren Partikelformen, welche von einem Menschen während der manuellen Annotation der Bilder nicht in einzelne Partikel trennbar sind. Da in der manuellen Segmentierung keine einzelnen Partikel des Agglomerats vorliegen, kann keine Gewichtungskarte für Pixel berechnet werden, die zwischen zwei Partikeln des Agglomerates liegen. Dadurch kann die KI nicht lernen, die Partikel zu trennen. Ein Sonderfall dieser Agglomerate sind überlagerte Plättchen. Flache Nanopartikel liegen besonders häufig auf einer ihrer großen Flächen auf dem Probenträger. Dadurch werden sie besonders leicht von sphärischen Partikeln überlagert. In diesem Fall sind die Erkennung und somit Trennung einzelner Partikel durch den Menschen zu realisieren. Dieser Fall trat im Trainingsdatensatz des Modells auf. Auch im Validierungsdatensatz gab es einige Fälle. Da überlagerte Partikel eine hohe Intensität zeigen, wirken die intensitätsbasierten Gewichtungskarten hier besonders stark. Die Trennung der überlagernden Partikel wurde dadurch begrenzt möglich.

Die Trennung von Partikel in Vordergrund und Hintergrund resultiert nicht in einer Klassifizierung ganzer Nanopartikel in einzelne Klassen. Sie ist jedoch ein wichtiger Schritt, um die Einteilung zu gewährleisten. Nur wenn die Segmentierung einer gegebenen Partikelverteilung in einzelne Partikel (Vordergrundobjekte) möglich ist, kann dieser Vordergrund von einer weiteren KI in eine Partikelklasse identifiziert werden. Die Hauptproblematik bilden hierbei weiterhin nicht trennbare Agglomerate in der Probe. Partikelverteilungen sind selten so homogen, dass sie nur aus einer Partikelform (z. B. Würfeln) bestehen. Kann das Modell die Agglomerate nicht voneinander trennen, werden nachfolgend mehrere Objekte als ein gemeinsames großes Objekt klassifiziert. Agglomeriert in einer Probe eine bestimmte Morphologie häufiger als andere Formen, beeinflusst dies zwar nicht zwangsläufig die gemessene Größenverteilung, jedoch die gemessene Formverteilung der Probe. Dies ist ein offenes Problem. Weiterhin muss davon ausgegangen werden, dass alle in einer Probe vorhandenen Partikelformen in gleichem Maße an der Agglomeration beteiligt sind. Andernfalls ist eine Analyse von Nanopartikeln von STEM-Aufnahmen schlichtweg nicht möglich.

Die Partikelgrößen wurden ebenfalls direkt aus der Segmentierung abgeleitet. Da bereits gezeigt wurde, dass die Einführung intensitätsbasierter Gewichtungskarten die Segmentierung des Modells signifikant verbessert, wurde die weiterführende Bestimmung der Partikelgrößen der Segmentierungen ausschließlich mittels eines intensitätsbasierten Modells durchgeführt. Für die Größenbestimmung der Partikel wurden die Partikelgrößen der Annotationen des Validierungsdatensatzes bestimmt und mit den Partikelgrößen aus den Segmentierungen verglichen. Die Bestimmung der Partikelgrößen erfolgte über die segmentierten Flächen der Partikel. Die Partikel wurden dabei nicht nach Bildern getrennt, um eine Auswertung zu vereinfachen. Die Trennung einzelner Partikel erfolgte nach der 4-Konnektivität. Das heißt, benachbarte Partikel durften sich nicht mit ihren Außenkanten berühren, um als getrennte Partikel betrachtet zu werden. Die Berechnung der Größe erfolgt durch die Berechnung des minimalen und maximalen Feret Durchmessers und des Äquivalenz Durchmessers. **Abbildung 31** zeigt die Größenverteilung aller im Validierungsdatensatz vorhandenen Partikel.

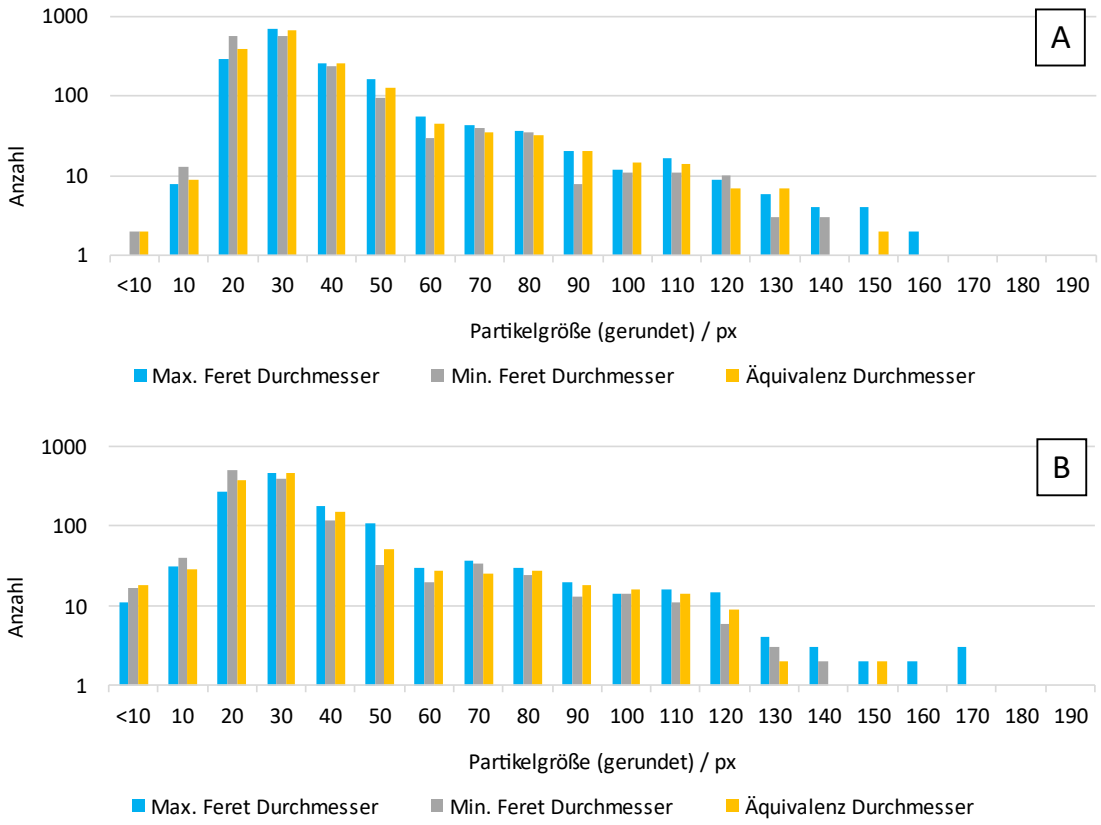


***Abbildung 31**: Größenverteilung aller Partikel in den vom Menschen erstellten Annotationsmasken des Validierungsdatensatzes (**A**) und den von dem Modell berechneten Segmentierungen (**B**). Die Partikelgrößen wurden ohne Berücksichtigung der Morphologie berechnet. Es ist zu erkennen, dass die Anzahl der vom Modell segmentierten Objekte geringer ist als in der menschlichen Annotation. In den menschlichen Annotationen wurden insgesamt 1645 Vordergrundobjekte identifiziert. Das Modell hingegen erkannte lediglich 1229 einzelne Objekte. Dies entspricht 75 % der vom Menschen identifizierten Partikel. Dies ist nicht dem Rand Index gleichzusetzen. Ein Teil der nicht identifizierten Partikel kann von dem Modell als Hintergrund erkannt worden sein. Hintergrund kann dann nicht in der Größenverteilung berücksichtigt werden. Wie in der Größenverteilung zu sehen, ist ein Teil der Objekte trotz geringen Rand Index in Agglomeraten vertreten. Auch wurden in der Analyse keine Objekte einbezogen, die am Rand des Bildes liegen beziehungsweise von diesem abgeschnitten werden. Die Anzahl der Objekte mit einem Durchmesser von <20 Pixeln nahm ebenfalls zu, was für die Segmentierung von wenigen Pixeln großen Objekten spricht. Dies ist ebenfalls ein fehlerhaftes Verhalten des Modells. (Die Analyse wurde ohne die Berücksichtigung eines Maßstabes vorgenommen, um die Berechnungen nicht zu verzerren.)*

Die Partikeldurchmesser wurden für den minimalen und maximalen Feret Durchmesser wie auch den Äquivalenzdurchmesser bestimmt. Die mittlere Partikelgröße betrug: (i) 34±22 Pixel in den Annotationen und 34±26 Pixel in den Segmentierungen, bezogen auf den maximalen Feret Durchmesser. (ii) 29±19 Pixel in den Annotationen und 28±20 Pixel in den Segmentierungen, bezogen auf den minimalen Feret Durchmesser; (iii) 31±20 Pixel in den Annotationen und 30±21 Pixel in den Segmentierungen, bezogen auf den Äquivalenzdurchmesser. Auf eine Skalierung in Nanometer wurde in diesem ersten Schritt verzichtet. Die Abweichung der Werte zwischen menschlicher Annotation und Segmentierung des Modells ist marginal. Eine automatische Auswertung hinsichtlich der Größe von Partikeln ist mit dem hier vorgestellten Modell verlässlich zu erreichen. Die Abweichung zwischen der Anzahl der erkannten Partikel und dem geringen Rand-Fehlers ist vermutlich der Methodik zur Berechnung des Randfehlers geschuldet. Die vom Modell erkannten Partikel werden zudem wahrscheinlich nicht alle echte Partikel sein. Wie in **Abbildung 30** zu erkennen ist, kann auch Hintergrund als Partikel erkannt werden. In einer fortgeschrittenen Analyse würde ein solcher Fehler berücksichtigt. In der einfachen Analyse der Partikelgrößen wie hier gezeigt, jedoch nicht. Die hier gezeigte Analyse diente daher zu ersten Einschätzung, ob das Model (a) in der Lage ist, STEM-Bilder zu segmentieren und (b) in der Lage ist, die mittlere Partikelgröße des Datensatzes zu bestimmen.

Da beide genannten Faktoren hinreichend erfüllt waren, wurde das auf intensitätsbasierten Gewichtungskarten trainierte Modell für die Analyse von Einzelbeispielen genutzt.

4.1.2 UNet++ trainiert mittels realer SE-Aufnahmen

Die Leistung des Modells zur Segmentierung von SE-Aufnahmen ist in **Tabelle 2** dargestellt. Nach den Erfolgen des Einsatzes intensitätsbasierter Gewichtungskarten in STEM-Bildern wurde dieser Ansatz auch in SE-Bildern angewendet. Für einen Vergleich zwischen distanzbasierten und intensitätsbasierten Gewichtungskarten wurden zwei unterschiedliche Netzwerke trainiert und mit dem gleichen Validierungsdatensatz getestet (**Tabelle 2**).

***Tabelle 2**: Auswertung der UNet++ Modelle für die Segmentierung von SE-Bildern, trainiert mittels gleichem Datensatz und unterschiedlichen Gewichtungskarten. Im Falle von SE-Bildern führt die Berücksichtigung der Bildintensität für die Berechnung der Gewichtungskarten zu keiner Verbesserung des Modells.* [81]

	IoU	Precision	Recall	Rand Index
Unet++ *Distanzbasierte Gewichtungskarten*	93 ± 2 %	97 ± 1 %	96 ± 5 %	2 ± 2 %
Unet++ *Intensitätsbasierte Gewichtungskarten*	91 ± 3 %	98 ± 2 %	93 ± 3 %	2 ± 16 %

Beide Modelle zeigten eine ähnliche Qualität. Die erreichte Gesamtleistung der Modelle ist geringer als die der STEM-Modelle. Zum einen wird hier die Komplexität der SE-Bilder eine Rolle spielen, zum anderen die Menge der Partikel auf den Bildern selbst. Im Falle von STEM-Bildern konnten nur REM-Bilder verwendet werden, die eine geringe Anzahl von überlappenden Partikeln aufwiesen. Überlappende Partikel (Agglomerate) konnte die KI nicht in Einzelpartikel segmentieren, da bereits der menschliche Analyst keine Agglomerate auftrennen konnte. Typische SE-Bilder enthielten bis zu mehrere Tausend Partikeln. Diese hohe Anzahl an Partikeln macht es daher vertretbar, dass die ein höherer Anteil von Partikeln fehlerhaft segmentiert wurde als in STEM-Aufnahmen. Für eine aussagekräftige Statistik reicht dies Segmentierung im Falle von SE-Bildern aus. Das SE-Modell, welches lediglich distanzbasierte Gewichtungskarten verwendet, weist eine leicht höhere *Intersection-over-Union* auf. Auch der Rand Index ist geringer als bei dem intensitätsbasierten Modell. Der Rand Index ist höher als bei dem Modell zu Segmentierung von STEM-Bildern. Es ist zu vermuten, dass die Verteilung der Pixel komplexer ist als in STEM-Bildern, wodurch SE-Bilder generell schwieriger zu segmentieren sind. Der Rand Index von 2 ± 2 % % entspricht einer Trennung

von ca. 95 von 100 Partikeln. Allerdings berücksichtigt der Rand Index jedoch nicht, inwiefern die Partikeldimension korrekt ist.

Aufgrund des geringeren Intensitätsunterschiedes zwischen Partikeln und Hintergrund (Probenträger) ist eine intensitätsbasierte Gewichtung offensichtlich nachteilig im Trainingsprozess. Das Netzwerk kann ohne die Einbindung der Bildintensität besser lernen, zwischen zwei Partikeln zu unterscheiden. In SE-Bildern können häufig Fresnel-Effekte an den Kanten und Ecken der Partikel auftreten. Der Fresnel-Effekt ist jedoch nicht an jedem Partikel zu beobachten und tritt in Abhängigkeit von Partikelform, Orientierung und Distanz zum Probenträger auf. Kanten, die durch den Fresnel-Effekt hervorgehoben werden, sind die hellsten Bereiche des Bildes. Durch die intensitätsbasierten Gewichtungskarten wird das Netzwerk dazu gezwungen, sich auf diese Bereiche zu „konzentrieren“, obwohl diese wenigen hellen Kanten nicht von entscheidender Rolle für die Segmentierung eines Bildes sind. Auch ist die Intensität der Pixel zwischen zwei Partikel anders als bei STEM-Aufnahmen. Häufig werden Bereiche des Partikel-zu-Partikel Überganges von einem „Schatten“ zwischen den Partikeln begleitet. Dieser Schatten lässt sich durch eine Schwächung der emittierten Elektronen erklären. Distanzbasierte Gewichtungskarten lenken die „Aufmerksamkeit“ des Modells dagegen klar auf die Bereiche zwischen Partikel. Diese erfahren eine höhere Gewichtung in der Verlustfunktion und somit „Aufmerksamkeit“ des Modells.

Die Intensität des Bildes zu nutzen, um die Segmentierung zu verbessern, erwies sich als nicht ausreichend. Um andere Einflüsse auf die Segmentierung zu identifizieren, wurden Partikel auf ihre spezifischen Eigenschaften hin untersucht. Hierfür wurden die Partikelpositionen und die Positionen aller Pixel, die zu einem Partikel gehören, aus den Annotationen genutzt. Pixel, die in der Annotation zu einem Partikel gehören, wurden dann aus dem dementsprechenden REM-Bild ausgeschnitten. Dadurch konnten die Pixelwerte jedes Partikels untersucht werden. Auch konnte so die Segmentierungsleistung für einzelne Partikel gewonnen werden. Die Korrelation aus Partikeleigenschaften und erreichter Leistung wurde genutzt, um Schwachstellen des Modells aufzudecken.

Für die Partikel wurde das Signal-zu-Rauschen-Verhältnis (*Signal to Noise Ratio, SNR*) [82], die mittlere Intensität des Partikels und die Größe des Partikels berechnet (**Abbildung 32**). Diese wurden anschließend mit der erreichten IoU des Partikels in einen Zusammenhang gesetzt. So konnten Einflüsse auf die Segmentierung ermittelt werden. Mittels der Untersuchungen zu

diversen Einflüssen auf die Segmentierung konnten kleinere Schwachpunkte der KI identifiziert werden. Die Größe der identifizierten Partikel zeigt keinen Einfluss auf die Leistung des Modells. Der Hauptteil der annotierten Partikel besitzt eine Größe von 100-1000 Pixel. Kleine Partikel fallen besonders auf. Da in der Annotation auch Objekte als Partikel (Vordergrund) gekennzeichnet wurden, die teilweise von anderen Partikeln verdeckt werden, waren manche Partikel nur wenige Pixel groß. Die in der Segmentierung identifizierten Bereiche lagen dann in einem Bereich von <10 Pixeln. Zudem sieht man in der **Abbildung 32-A** den Fehler, die von der Erstellung des Datensatzes herrühren. Einige Partikel besitzen eine Größe von einem Pixel. Dies sind „Geisterpartikel", also Fehler in der Annotation, die nun als echtes Partikel gewertet wurden. Sie zeigen, dass ein menschlicher Fehler in solchen Datensätzen präsent ist. Der Einfluss des Rauschens ist in **Abbildung 32-B** gezeigt. Als Metrik wurde das Signal-zu-Rauschen-Verhältnis genutzt (*Signal to Noise Ratio, SNR*). Es wird nach **Abbildung 32** berechnet.

$$SNR = \frac{\mu_{Bild}}{\sigma_{Bild}} \qquad 31$$

Das SNR beschreibt die mittlere Pixelintensität eines Objektes, dividiert durch die Standardabweichung der Pixelintensität. Eine Pixelgruppe mit einer Standardabweichung von 0 entspricht einer einfarbigen Fläche. Je höher das SNR liegt, desto geringer ist das Rauschen innerhalb der Partikel. Die mittlere Partikelintensität μ_{Bild} aller im Validierungsdatensatz vorhandenen Partikel beträgt 125 mit einer Standardabweichung σ_{Bild} von 10,4. Gemäß **Formel 31** kann das mittlere SNR des Datensatzes berechnet werden. Dieses liegt bei $SNR = 12$. Interessanterweise zeigt das SNR einen geringen Einfluss auf die Segmentierung der Partikel. Die berechnete Regressionsgrade in **Abbildung 32-B** besitzt einen R^2 Wert von 0.01. Dies widerspricht im geringen Maße der Funktionsweise von Autoencodern, zu denen auch die verwendete UNet++ Architektur gehört. Innerhalb des kontrahierenden Parts eines Autoencoders werden die Eigenschaften eines Bildes so weit kodiert, dass das Bildrauschen ignoriert werden sollte. [63, 83] Dennoch ist der Einfluss des Rauschens minimal vorhanden.

Als letzten möglichen Einfluss auf die Segmentierung eines SE-Bildes wurde die mittlere Intensität der in einem Bild vorhandenen Partikel untersucht. Die mittlere Intensität eines Partikels wurde bereits genutzt, um das SNR des Partikels zu berechnen und soll hier noch einmal gesondert betrachtet werden. Wie bereits in **Abbildung 23** (S. 56) dargestellt, existiert

ein Intensitätsgradient zwischen übereinanderliegenden Partikeln. Partikel mit einer geringen mittleren Intensität sind Partikel, die oftmals von anderen Partikeln überlagert werden. Dementsprechend sind diese „Hintergrund-Partikel“ von anderen Partikeln abgeschnitten und nicht vollständig abgebildet. Elongierte Partikel wie Stäbchen sind zudem häufig durchschnitten. Das bedeutet, dass zwei übereinanderliegende Stäbchen durch die Annotation in ein ganzes Stäbchen (oberes Partikel) und zwei Teil-Stäbchen (unteres Partikel) getrennt werden. Die Auswertung des Bildes auf Seite 155 zeigt die Problematik durchschnittener Stäbchen auf. Die tendenziell weiter unten liegenden Partikel werden nach den Ergebnissen aus **Abbildung 32-C** schlechter segmentiert. Dies wirkt sich zwar negativ auf die IoU aus, ist in der Anwendung aber nicht von Belangen, da solche „Hintergrund-Partikel“ aussortiert werden.

Zusammenfassend kann gesagt werden, dass die Größe einzelner Partikel keinen Einfluss auf ihre Segmentierung hat, während mittlere Intensität und Rauschen eines Partikels einen geringen Einfluss auf die Segmentierung besitzen. Eine Modifizierung des Modells bzw. der Trainingsroutine, basierend auf den Ergebnissen aus **Abbildung 32,** wurde jedoch nicht vorgenommen. Der Einfluss der gemessenen Parameter auf die Leistung des Modells war zu gering.

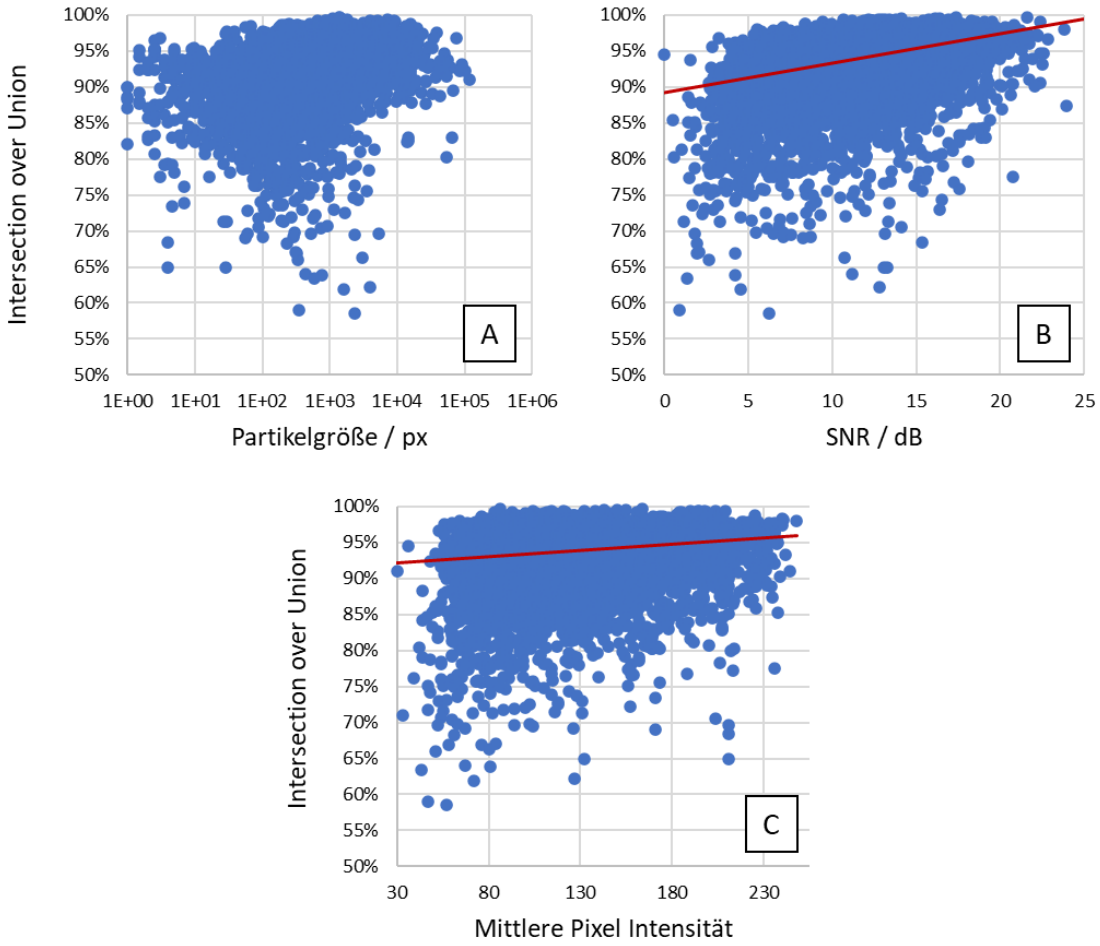


***Abbildung 32**: Einfluss verschiedener Partikeleigenschaften auf die Segmentierung individueller Nanopartikel. Gezeigt sind ca. 9.000 der 15.000 Partikel aus den Bildern des SE-Validierungsdatensatzes. Die Partikel wurden zufällig ausgewählt. Eine Reduktion des Datensatzes war aus technischen Gründen nötig. Aus der Darstellung lässt sich entnehmen, dass die Größe des Partikels keinen Einfluss auf die Segmentierung nimmt. Beim Signal-zu-Rauschen-Verhältnis nimmt die mittlere Leistung des Netzwerkes mit zunehmendem SNR jedoch zu. Der Einfluss ist jedoch gering, wie der lineare Fit (rote Linie) mit einem R^2=0.01 zeigt. Auch nimmt die Segmentierung mit zunehmender mittlerer Intensität zu. Hier zeigt der lineare Fit mit R^2=0.017 zwar eine leicht bessere Anpassung, ist jedoch ebenfalls zu gering, um eine statistisch relevante Aussage zu treffen.*

Wie bereits in den STEM-Aufnahmen lässt sich nach der erfolgreichen Segmentierung die mittlere Partikelgröße direkt aus den Segmentierungen ableiten. **Abbildung 33** zeigt den die Partikelgrößenverteilung für Partikel aus den händisch erstellten Annotationen und denen aus den vom Modell erstellen Segmentierungen. Die Größenverteilung der Segmentierungen zeigt

denselben Trend wie die Größenverteilung der Annotation. Das Modell konnte hingegen nur ca. 13.000 der 15.000 im Datensatz markierten Partikel identifizieren. Dies entspricht einer Rate von 87 %. Die Betrachtung der Partikelgrößenverteilung lässt den Schluss zu, dass zwar weniger Partikel von dem Modell erkannt werden, dies aber größenunabhängig geschieht. Tendenziell werden unabhängig von der Größe alle Partikel gleich gut segmentiert.

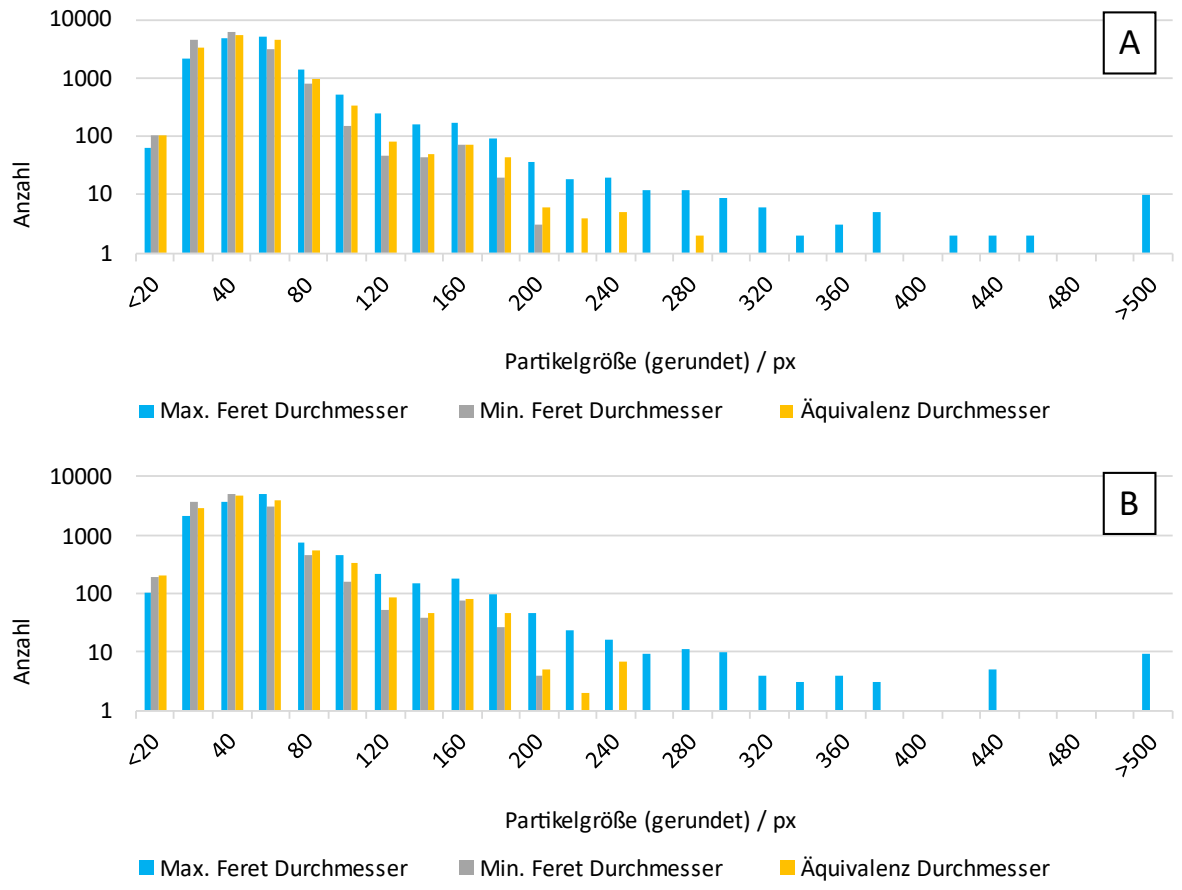


***Abbildung 33**: Auswertung aller im SE-Validierungsdatensatz enthaltenden Partikel. Die Grafik zeigt die Verteilung der Partikelgrößen in den händisch erstellten Annotationen (**A**) gegenüber den Partikelgrößen, gemessen in den Segmentierungen des Modells (**B**). Die Partikel sind nicht nach ihrer Form klassifiziert oder sortiert. In den Annotationen wurden insgesamt ca. 15.000 einzelne Partikel markiert. Das Modell war in der Lage, ca. 13.000 Partikel zu lokalisieren.*

Die Partikeldurchmesser wurden für den minimalen und maximalen Feret Durchmesser wie auch den Äquivalenzdurchmesser bestimmt. Die mittlere Partikelgröße betrug: (i) 47±39 Pixel in den Annotationen und 46±40 Pixel in den Segmentierungen bezogen maximalen Feret Durchmesser. 33±21 Pixel in den Annotationen und 32±22 Pixel in den Segmentierungen, bezogen auf den minimalen Feret Durchmesser; (iii) 37±23 Pixel in den Annotationen und

36±24 Pixel in den Segmentierungen, bezogen auf den Äquivalenzdurchmesser. Auf eine Skalierung in Nanometer wurde in diesem ersten Schritt verzichtet. Die Abweichung der mittleren Partikelgröße, die aus den Segmentierungen aller im Datensatz vorhanden Validierungs-Bilder, bestimmt werden konnte, war nicht signifikant. Die Partikelgröße kann mit ausreichender Sicherheit aus den Segmentierungen des Modells bestimmen werden.

Die mittlere Abweichung des Partikeldurchmessers der Segmentierung zur Annotation betrug ca. 1 Pixel. Die Vergrößerung des REMs besitzt daher einen Einfluss auf die gemessenen Partikeldurchmesser. Bei sinkender Vergrößerung wird der Maßstab (nm px^{-1}) zunehmen. Aufgrund der gemessenen Abweichung der mittleren Partikelgröße der Annotationen zu dem mittleren Durchmesser der Segmentierung kann eine lineare Abhängigkeit des Fehlers einer beliebigen Partikelgröße in Abhängigkeit zur Vergrößerung des Partikels prognostiziert werden. Partikel in einer geringen Vergrößerung, wie z. B. Mikrostäbchen werden einen höheren absoluten Fehler aufweisen als z. B. Nanokugeln.

Interessanterweise zeigten sich systematische Fehler in den Ergebnissen zwischen der KI und verschiedenen menschlichen Analysten. Dies zog die Frage nach sich, ob und wie stark die KI einen Bias aus den Trainingsdaten erlernt hat. Der Bias benennt in diesem Fall die Ausdehnung der Partikel in einer REM-Abbildung. Da die Partikelränder häufig diffus sind, ist der Partikeldurchmesser lediglich ein Schätzwert. Der Bias ist dabei der systematische Messfehler zwischen den gemessenen Partikelgrößen von 2 unterschiedlichen Analysten. In der weiteren Einzelanalyse von SE-Bildern wurden hierfür verschiedene Bilder von diversen Analysten untersucht.

Grundsätzlich war die Segmentierung von SE-Bildern mittels UNet++ verlässlich im Bezug auf die gemessenen Partikelgrößen. Eine Beeinträchtigung durch einen Bias in den Trainingsdaten könnte durch die Erstellung von Annotationsmasken verschiedener Analysten bereinigt werden. Dies bedeutet, dass für jedes Bild des Datensatzes mehrerer Personen eine Annotationsmaske anfertigen und anschließend lediglich die Pixel als Vordergrund (Partikel) markiert werden, die in jeder einzelnen Annotationsmaske als Vordergrund markiert wurden. Ob und wie stark sich dieser Mehraufwand in einer Verbesserung der KI niederschlägt, wurde in dieser Arbeit nicht weiterverfolgt.

4.2 Klassifizierung von Nanopartikeln

Die Klassifizierung von Nanopartikeln beschreibt die Einteilung von einzeln abgebildeten Nanopartikelbildern in eine zuvor vom Menschen definierte Klasse. Die Klassen beschreiben die Morphologie des gezeigten Nanopartikels. Die Benennung der Klassen erfolgte nach den Empfehlungen von Muñoz-Mármol. [64] Im Falle von STEM-Bildern entspricht die Benennung dem 2D Pendant zur erwarteten 3D Morphologie.

Da Klassifizierungsmodelle wie aus Segmentierungsmodelle eine Wahrscheinlichkeits-verteilung über alle verfügbaren Klassen berechnet, wurde das argumentative Maximum der jeweiligen Verteilung genutzt, um die finale Klasse bzw. Morphologie zu bestimmen. **Formel 32** & **33** zeigen dies für die *One Hot Kodierung* auf:

$$y_i = argmax(y_i) \tag{32}$$

$$\begin{pmatrix}0\\0\\1\end{pmatrix} = argmax\begin{pmatrix}0{,}03\\0{,}06\\0{,}91\end{pmatrix} \tag{33}$$

Die Hauptmetrik zur Messung der Leistung eines Modells ist die Korrektklassifizierungsrate (*Accuracy*). Sie ergibt sich wie bereits die *Intersection-over-Union* aus der gezeigten Konfusionsmatrix (**Abbildung 29**, S. 71). Die Accuracy entspricht dem relativen Anteil der vom Modell korrekt klassifizierten Partikel:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{34}$$

Vor dem Training wurden die Datensätze gesplittet, wobei der kleinere Teil zur Validierung des Modells bereitgestellt wurde. Diese Bilder wurden ausschließlich für die Validierung genutzt. Das Netzwerk hat daraus keine Eigenschaften lernen können. Validierungsdatensätze wurden für SE- und STEM-Bilder separat erstellt. Die gezeigten Ergebnisse wurden mittels dieser Datensätze gewonnen.

4.2.1 Partikelklassifizierung mittels AlexNet (STEM)

Die Klassifizierung von Nanopartikeln aus STEM-Bildern mittels AlexNet wurde anhand des Validierungsdatensatzes getestet. Der Datensatz enthielt 1425 individuelle Partikel in 8 Klassen. Die Ergebnisse sind in **Tabelle 3** dargestellt.

***Tabelle 3**: Ergebnisse der Validierung des Klassifizierungsnetzwerkes für Nanopartikel aus STEM-Bildern. Es zeigt sich eine kontinuierlich hohe Klassifizierung der KI. Die Auswertung des Klassifizierungsnetzwerkes erlaubt keine Darstellung von Konfidenzintervallen.*

Klasse	Agglomerat	Hintergrund	Kreis	Sechseck	Fünfeck	Viereck	Stäbchen	Dreieck	Gesamt
Anzahl	194	182	287	117	102	154	102	287	1425
Accuracy	93 %	100 %	92 %	98 %	97 %	96 %	93 %	95 %	95 %

Es zeigt sich, dass das Netzwerk in der Lage ist, die gezeigten Partikel mit hoher Korrektheit zu klassifizieren. Die Klassifizierung hat jedoch eine Schwachstelle, welche im Datensatz selbst liegt. Aufgrund der Einteilung der Partikel in Klassen hat sich wahrscheinlich der Bias des Autors dieser Arbeit in dem Datensatz festgesetzt. Die KI vertritt die Meinung des Autors dieser Arbeit bezüglich der Partikelformen. Um zu überprüfen, inwiefern sich die Meinung von Menschen bezüglich der Auswertung von Nanopartikeln unterscheidet, wurde eine Umfrage unter chemisch erfahrenen Personen des Arbeitskreises Epple durchgeführt. Der erste Teil der Umfrage bestand aus der Einteilung diverser Partikel in eine der oben genannten Klassen. Dabei wurde festgestellt, dass die mittlere Sicherheit, mit welcher eine menschliche Gruppe ein Partikel einer Form zuordnet, 75 % beträgt. [69] Die Umfrage wurde mit insgesamt 23 Personen durchgeführt, die 35 Bilder einer STEM-Morphologie zuordnen mussten.

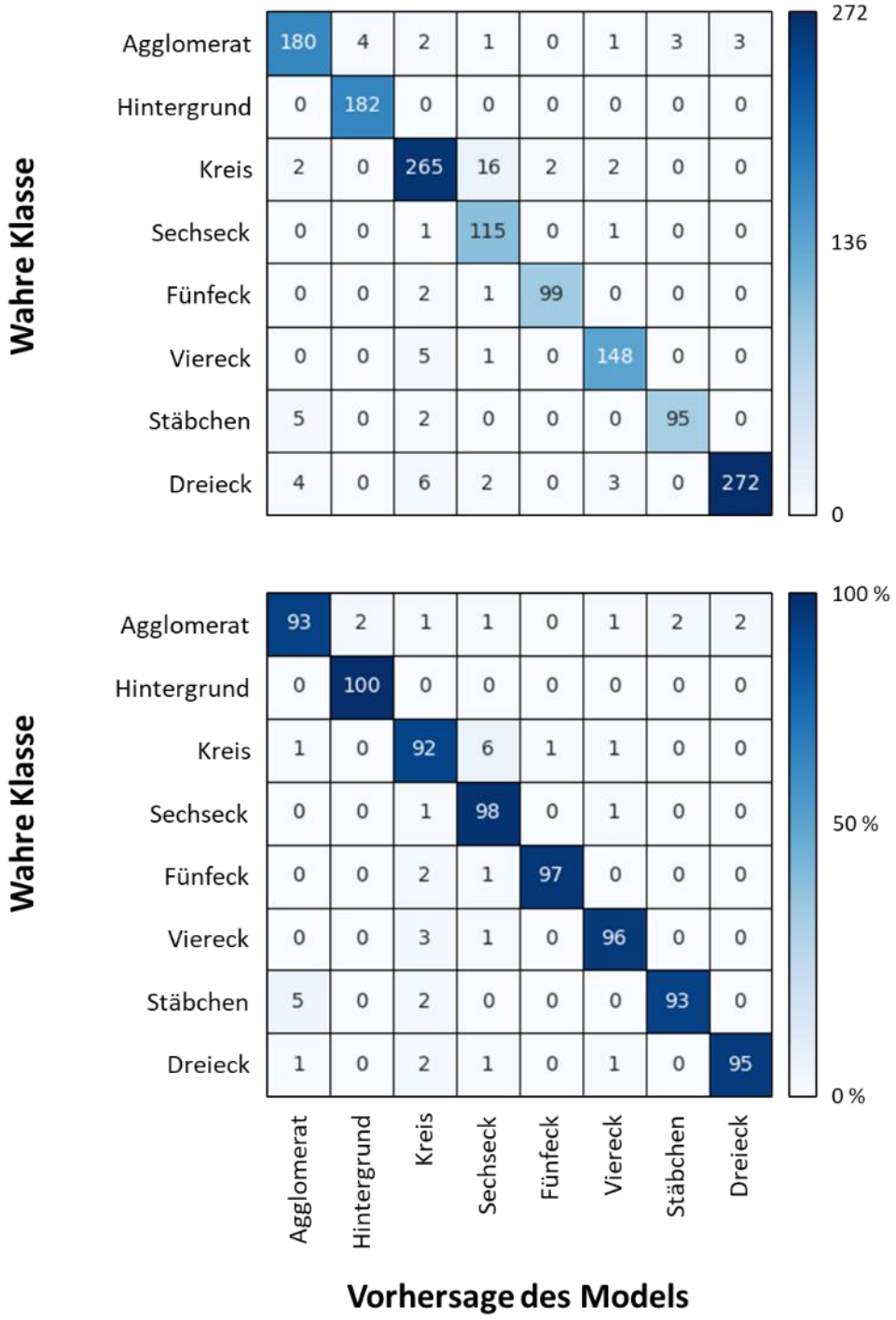


***Abbildung 34**: Konfusionsmatrizen zur Auswertung von 1425 Einzelpartikeln des Validierungsdatensatzes. Die Konfusionsmatrizen zeigen die Korrelation der wahren Klasse (menschliche Einschätzung) gegenüber der Einschätzung des Modells. Sie zeigen eine kontinuierlich hohe Präzision in allen Klassen. Die geringste Leistung wurde in den Klassen Stäbchen, Agglomerat und Kreis erreicht. Agglomerate und Stäbchen sind aufgrund ihrer Größe in den Bildern oftmals von weiteren Partikeln umgeben. Diese Partikel zeigen einen Einfluss auf die Klassifizierung des Modells. Kreise und Sechsecke waren tendenziell schlecht zu*

unterscheiden. Zudem können menschliche Fehler in der Erstellung des Datensatzes nicht ausgeschlossen werden.

Zudem wurde eine detailliertere Umfrage bezüglich Partikelformen mit 5 Personen durchgeführt, die ca. 1.300 Partikel in 20 verschiedenen STEM-Bildern auswerten sollten. Die Umfrage sollte in einem breiteren Maßstab klären, inwiefern sich die Auswertung diverser Analysten unterscheidet. Die Ergebnisse sind in **Tabelle 4** dargestellt. Zuzüglich zu den Morphologie-Klassen, die im KI-Training verwendet wurden, konnten die menschlichen Analysten Partikel auch als „Unsicher" bezeichnen. Dies sollt massenhaftes Raten einer Morphologie verhindern.

***Tabelle 4**: Analyse von ca. 1300 Nanopartikeln bezüglich ihrer Form. Die Analyse wurde von 5 unterschiedlichen menschlichen Analysten durchgeführt. Insgesamt wurden 20 unterschiedliche Bilder ausgewertet. Tabelle entnommen und abgewandelt nach Bals et al. [69]*

	Kreise	Sechseck	Fünfeck	Viereck	Stäbchen	Dreieck	Unsicher	Summe
Analyst I	266 (21 %)	23 (2 %)	26 (2 %)	134 (10 %)	44 (3 %)	177 (14 %)	627 (48 %)	1297
Analyst II	251 (20 %)	74 (6 %)	26 (2 %)	132 (10 %)	23 (2 %)	131 (10 %)	674 (51 %)	1318
Analyst III	145 (10 %)	7 (<1 %)	55 (4 %)	143 (10 %)	30 (2 %)	104 (7 %)	917 (65 %)	1401
Analyst IV	607 (44 %)	52 (4 %)	21 (2 %)	199 (15 %)	36 (3 %)	126 (9 %)	328 (24 %)	1369
Analyst V	309 (24 %)	162 (20 %)	93 (7 %)	290 (22 %)	26 (2 %)	101 (8 %)	329 (25 %)	1310
Durchschnitt	317 ± 155 (24 ± 12 %)	64 ± 54 (5 ± 4 %)	44 ± 27 (3 ± 2 %)	180 ± 60 (13 ± 4 %)	32 ± 7 (2 ± 1 %)	128 ± 27 (10 ± 2 %)	575 ± 224 (43 ± 17 %)	1340 ± 39

Aus **Tabelle 4** ist zu entnehmen, dass deine Gruppe von menschlichen Analysten diverse Ergebnisse bezüglich einer gegebenen Partikelverteilung in STEM-Bildern liefert. Die Problematik in Bezug auf das Training einer KI zur Klassifizierung ist, dass der Datensatz, welcher von einem (1) Analysten erstellt wurde, wahrscheinlich von anderer Seite abgelehnt werden würde. Eine Lösung des Problems wäre, einen Datensatz zu erstellen, der von mehreren Analysten aus einer zufälligen Menge von Nanopartikeln erstellt wurde. Jeder Analyst würde für sich einen Datensatz aus ca. 1000-5000 Nanopartikeln pro Klasse erstellen.

Die Partikel, die in jedem Datensatz in derselben Klasse auftreten, würden dann in einen gemeinsamen Datensatz aufgenommen werden. Dieser Prozess ist sehr zeitaufwendig und daher kaum realisierbar.

Unter Zuhilfenahme einer einfachen Fehlerbetrachtung konnte die zu erwartende Abweichung des Klassifizierungsnetzwerkes zu einem beliebigen Analysten berechnet werden. Die mittlere Abweichung zwischen verschiedenen menschlichen Analysten betrug ca. 22 % (**Tabelle 4**). Die Abweichung zwischen dem Autor des STEM-Datensatzes und der KI liegt bei 5 %. Die zu erwartende Abweichung zwischen der KI und einem beliebigen Menschen liegt demnach abgeleitet bei ca. 27 % (22 % + 5 %).

Weiterhin wurde im Zuge der menschlichen Auswertung die Frage gestellt, worauf die KI achtet, wenn diese Bilder klassifiziert. Daher wurde das Modell der Analysemethode von Zeiler & Fergus unterzogen. [27] In diesem Verfahren wurden die Pixel des Bildes mittels Gradientenabstiegsverfahren iterativ angepasst, um das neuronale Netz maximal anzusprechen (**Abbildung 35**). Die so generierten Bilder zeigen die Bereiche des Bildes mit hoher Intensität, die auf die Entscheidung des neuronalen Netzes den größten Einfluss zur Klassifizierung haben.

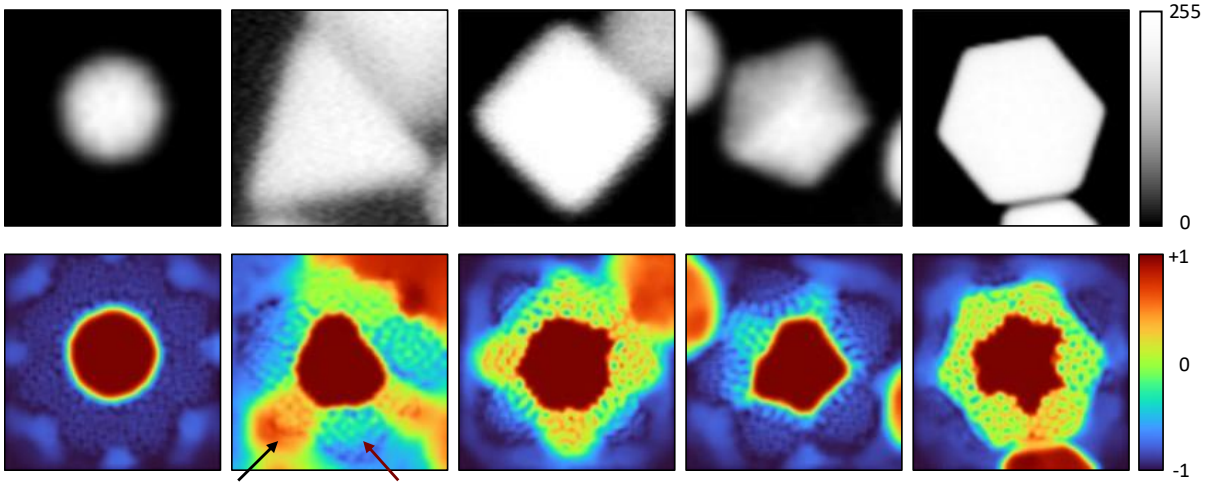


***Abbildung 35**: Interpretation ausgewählter Trainingsbilder nach dem Prinzip von Zeiler & Fergus. [27] Die gezeigten Bilder wurden so verändert, dass sie Bereiche sichtbar machen, die das neuronale Netz in der Entscheidungsphase als „wichtig" erachtet. Das Bild wurde anschließend auf das Intervall [-1, 1] normalisiert. Die Ecken der Partikel (schwarzer Pfeil) zeigen im Vergleich zu den Kanten eines Partikels (roter Pfeil) hohe Beachtung. Aufgrund der Skalierung auf 224x224 Pixel ist die absolute Größe der Partikel ohne Belangen.*

Die höchste Intensität erreicht das Partikel selbst. Es zeigt sich weiterhin, dass die KI ähnlich zu einem menschlichen Analysten den Ecken und Kanten wechselnde Aufmerksamkeit schenkt. Spitzere Ecken von dreieckigen Partikeln zeigen dabei die auffälligste Intensität. Kanten besitzen tendenziell eine geringere Intensität. Dieser Trend nimmt mit zunehmender Anzahl der Ecken ab, bis schließlich nur das Partikel selbst eine hohe Intensität aufweist. Somit konnte gezeigt werden, dass die KI die Bilder auf ähnliche Art auswertet wie ein Mensch: *Es zählt Ecken und Kanten.*

Aufgrund der Abweichungen zwischen den menschlichen Analysen wurde eine unabhängige Untersuchung des Datensatzes bezüglich der Unterscheidbarkeit der Bilder durchgeführt. Dies sollte auch die Komplexität der STEM-Bilder aufzeigen. Hierfür wurde eine vortrainierte Version von VGG19 verwendet. VGG19 ist ein kompaktes neuronales Netz. Dies ist auf den ImageNet-Datensatz vortrainiert worden. [84] Dadurch konnte einen ein menschlicher Bias ausgeschlossen werden, da der Datensatz von vielen Menschen erstellt wurde. Zum anderen enthält der ImageNet-Datensatz eine Vielzahl unterschiedlicher Objekte, wodurch ein breiteres Spektrum an allgemeinen Bildeigenschaften abgedeckt wurde. Anschließend wurden die einzelnen Partikel nach der Methode von van der Maaten visualisiert. [85] Partikel, die dabei unterscheidbarer voneinander waren, sind weiter voneinander geplottet als solche, die wenige Eigenschaften miteinander teilen. **Abbildung 36** zeigt, wie sich verschiedene Formen leicht voneinander abgrenzen. Der generelle Trend zeigt, dass die Partikel durcheinander vorliegen. Dies spricht für ein undefiniertes Auftreten von allgemeinen Bildeigenschaften. Stattdessen bilden STEM-Bilder und die darin abgebildeten Nanopartikel komplexe Objekte mit spezifischen Bildeigenschaften, die eine spezifisches KI-System und Datensatz nötig machen.

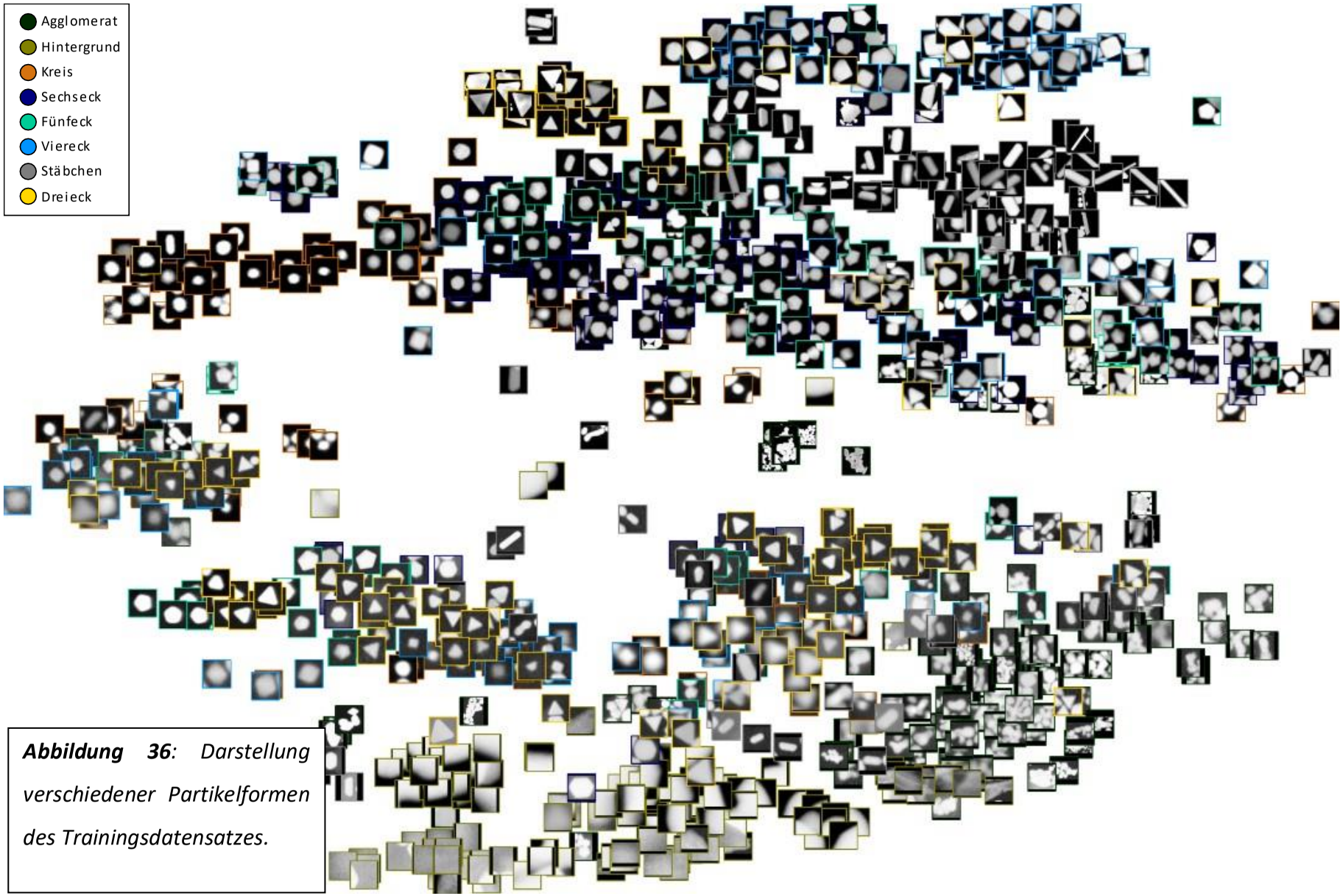


***Abbildung 36**: Darstellung verschiedener Partikelformen des Trainingsdatensatzes.*

***Abbildung 36-Fortsetzung**: Die Abbildung zeigt die Ergebnisse der unabhängigen Analyse von 128 Partikeln pro Klasse und die relative Nähe der Partikeleigenschaften zueinander. Je näher die Eigenschaften zweier Partikel, desto näher werden diese zueinander geplottet. Es ist ersichtlich, dass die Bild-Eigenschaften der Partikelklassen sehr nahe liegen, die geplotteten Partikel liegen somit häufig überlappt vor. Die unabhängige Analyse des Datensatzes bestätigt die Auswertung von mehreren menschlichen Analysten, dass die Partikelform häufig mehrdeutig interpretiert werden kann. Selbst ein leistungsstarkes KI-System unterliegt dieser Problematik. Dies bestätigt auch die Notwendigkeit des Trainings eines spezifischen Modells zur Erkennung der Partikelformen in STEM-Bildern. Es muss allerdings auch angemerkt werden, dass die Partikel sehr dominant nach ihrem Kontrast eingeteilt wurden. Das in dieser Arbeit trainierte neuronale Netz ist vermutlich auch auf diese Kontrastunterschiede spezialisiert, um Partikel zu unterscheiden.*

4.2.2 Partikelklassifizierung mittels ResNet34 (SE)

Die Klassifizierung von Nanopartikeln aus SE-Bildern mittels ResNet34 wurde anhand des Validierungsdatensatzes getestet. Der Datensatz enthielt 3469 individuelle Partikel in 5 Klassen. Die Ergebnisse sind in **Tabelle 5** dargestellt.

***Tabelle 5**: Validierung des Netzwerkes zur Klassifikation von Nanopartikeln aus SE-Bildern. Die Validierung zeigt, dass die Leistung des Netzwerkes konstant hoch ist. Lediglich Kugeln und kugelartige Partikel zeigen Ergebnisse mit einer Korrektklassifizierungsrate unter 90 %. Die liegt vermutlich an der Ähnlichkeit dieser beiden morphologischen Ausprägungen.*

Klasse	Hintergrund	Würfel	Stäbchen	Kugel	Kugelartig	Gesamt
Anzahl	367	704	474	1001	923	3469
Accuracy	94 %	96 %	99 %	86 %	89 %	91 %

Das Netzwerk ist in der Lage, diverse Partikelformen akkurat zu klassifizieren. Die erreichten Korrektklassifizierungsraten entsprechen denen eines Menschen und reichen sogar über die menschliche Leistung hinaus. [69] Wie bereits in dem Datensatz für STEM-Partikel liegt auch in diesem ein starker Bias des Autors vor. Um den Bias zu überprüfen, wurde jedoch keine weitere Umfrage durchgeführt, da die zu erwartende Abweichung zwischen der Klassifizierung der KI und einem beliebigen Menschen bereits aus den STEM-Daten hervorgeht. Die Abweichung der KI zum Menschen beträgt 9 % (**Tabelle 5**). Ähnlich zu den STEM-Daten beträgt die zu erwartende Abweichung der KI zu einem beliebigen Menschen ca. 31 % (22 % + 9 %).

Die Konfusionsmatritzen zeigen die detaillierte Verteilung der fehlerhaft klassifizierten Partikel (**Abbildung 37**). Die höchste Verwechselung findet zwischen Kugeln und Kugelartigen statt. Dies liegt in der unklaren Definition von kugelartigen Partikeln bzw. Kugeln. Während der Erstellung des Datensatzes wurde darauf geachtet, nur „perfekt“ runde Partikel als Kugeln zu definieren. Durch die Natur eines REM-Bildes (Rauschen, Verzerrung, Aufladungseffekte) war es allerdings nicht immer möglich, diese Einteilung zu befolgen. Kugelartige Partikel sind hingegen ein Sammelbecken für deformierte, elliptische und unklare Partikelformen. Hier hinein fallen alle Partikel, die sich nicht klar von anderen Formen abgrenzen lassen. Auch stark

abgeflachte Partikel (gerundete Plättchen, Dodekaeder, Würfel & Oktaeder) wurden in diese Kategorie hinzugezählt.

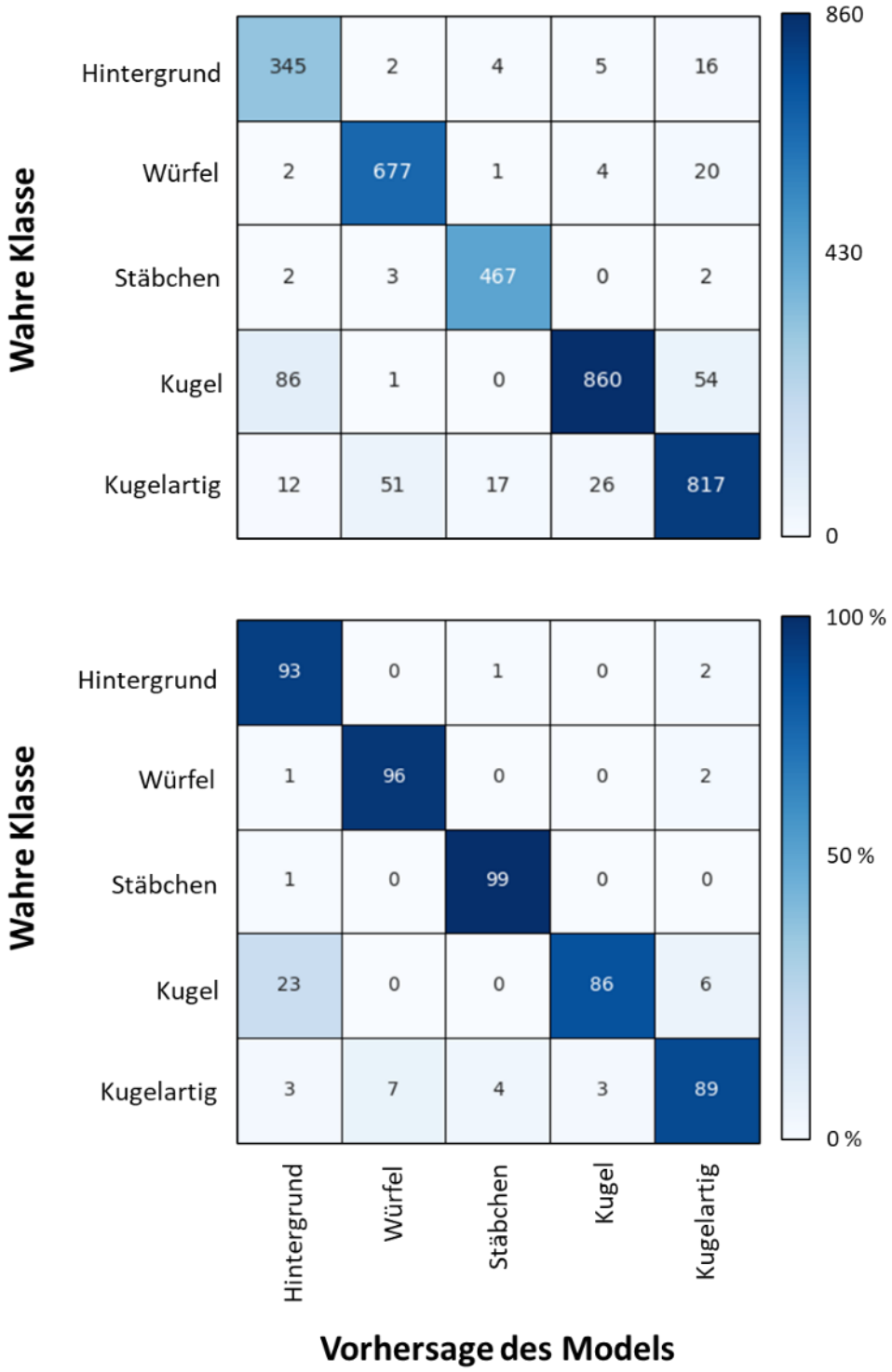


***Abbildung 37**: Konfusionsmatrizen zur Auswertung von 1425 Einzelpartikeln des Validierungsdatensatzes. Die Konfusionsmatrizen zeigen die Korrelation der wahren Klasse (menschliche Einschätzung) gegenüber der Einschätzung des Modells. Es wird eine kontinuierlich hohe Präzision in allen Klassen erreicht. Schwach ist hingegen die Einteilung von*

Kugeln und Kugelartigen. Die hier gezeigte Klasse Hintergrund deckt nicht nur wahren Hintergrund (Probenträger) ab, sondern auch verdeckte Partikel Wie bereits im STEM-Datensatz können menschliche Fehler in der Erstellung des Datensatzes nicht ausgeschlossen werden.

Wie bereits für STEM-Partikel wurde eine unabhängige Untersuchung des Datensatzes bezüglich der Unterscheidbarkeit der Bilder durchgeführt. Erneut wurde eine vortrainierte Version von VGG19 verwendet. Dies bedeutet, dass dieselben Eigenschaften des Datensatzes untersucht wurden. Anschließend wurden die einzelnen Partikel nach der Methode von van der Maaten visualisiert. [85] Partikel, die dabei gemeinsame Eigenschaften aufwiesen, wurden näher zueinander geplottet als solche, die wenige Eigenschaften miteinander teilten. **Abbildung 36** zeigt, wie sich verschiedene Formen leicht voneinander abgrenzen. Der generelle Trend zeigt, dass die Partikel in Gruppen vorliegen. Dies spricht für einen gemeinsamen Trend der Bildeigenschaften innerhalb einer Klasse.

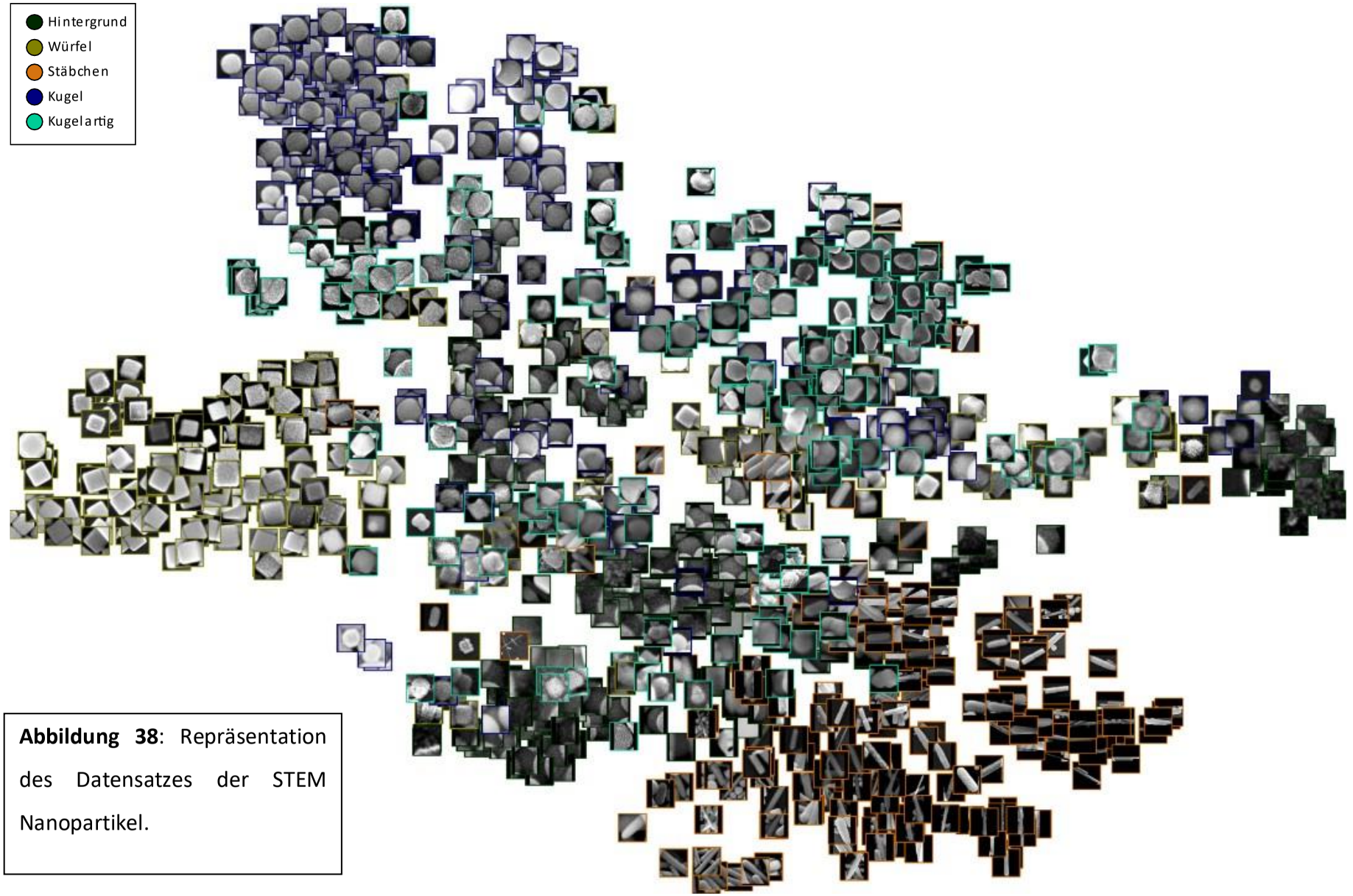


Abbildung 38: Repräsentation des Datensatzes der STEM Nanopartikel.

Abbildung 38 – Fortsetzung: Die unabhängige Analyse des Datensatzes mittels VGG19 zeigt die Differenz der Eigenschaften der einzelnen Klassen gut sichtbar auf. Die Klassen Stäbchen (orange), Würfel (gelbgrün) und Kugeln (blau) besitzen eine enge Eigenschaftsverteilung, wodurch diese nahe beieinander geplottet werden. Partikel der Hintergrund-Klasse (dunkelgrün) zeigen ebenfalls eine Tendenz, sich von den anderen Klassen abzugrenzen. Kugelartige Partikel (türkis) sind hingegen überall zu finden. Ihre Eigenschaften differenzieren sich nicht so stark von denen der anderen Klassen. Somit konnte unabhängig gezeigt werden, dass Partikel dieser Klasse generell schwer zu klassifizieren ist und die geringe Korrektklassifizierungsrate für kugelartige Partikel nicht eine Schwäche des verwendeten Modells darstellt.

4.3 Erzeugung künstlicher SE-Bilder mittels CycleGAN

Um das Problem des menschlichen Bias in den Datensätzen zu minimieren, wurde ein Ansatz gewählt, welcher vollständig auf menschliche Daten verzichtet. Vollständig künstliche Daten sollten dann genutzt werden, um die bisherigen KI-Systeme fehlerfrei zu trainieren.

Blender bietet die Möglichkeit, einfache Simulationen 3-dimensionaler Objekte zu generieren. **Abbildung 39** zeigt schrittweise die Simulation des Falls der Nanopartikel auf einen planaren Probenträger. Die erzeugten Landschaften aus Partikeln zeichnen ein deutliches Relief auf (**Abbildung 40**).

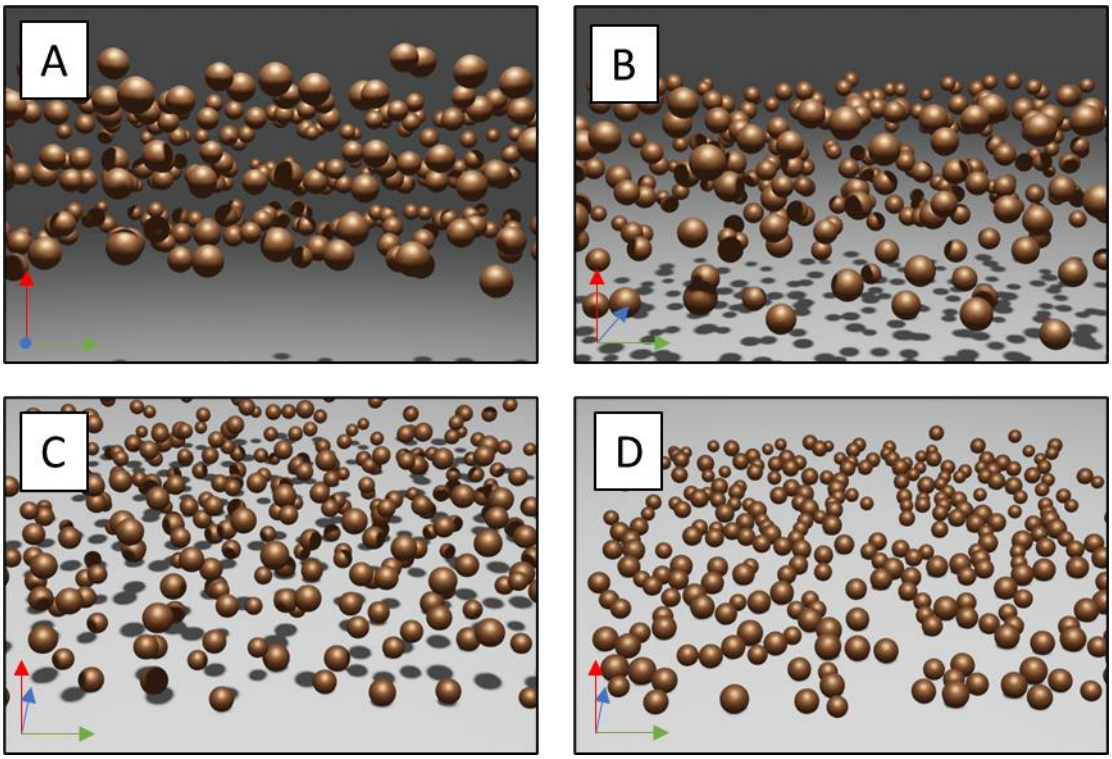


***Abbildung 39**: Fall der "Nanopartikel" auf einen planaren Probenträger. Die Partikel beginnen die Simulation in schwebender Position. Sie sind in mehreren Schichten angeordnet und jedes Partikel wurde um einen geringen zufälligen Vektor verschoben (**A**). Die Partikel lösen dann aus dieser Position und fallen auf den Probenträger (**B-C**) bevor sie auf diesem relaxieren und schließlich stoppen (**D**). Pfeile zeigen die Orientierung der Achsen in diesem Beispiel, da sich die Kamera ebenfalls bewegt. Für diese Simulation wurden 300 unterschiedlich große Kugeln verwendet.*

Da es sich bei Blender um eine Software zur Bearbeitung von Videomaterial handelt, war eine direkter Export der Oberflächendaten nicht zu verwirklichen. Stattdessen war es möglich, die

Distanz von der genutzten „Kamera“ zu jeder Fläche der Szenerie zu berechnen und anschließend linear invertiert auszugeben. Diese Höhenkarten wurden normalisiert für das Training des GANs genutzt. Somit konnten relative Höhendaten gewonnen werden. Auch die Umwandlung in absolute Höhenkarten ist unter Berücksichtigung des Maßstabes der generierten Bilder möglich. Da Bilder für das KI-Training jedoch normalisiert vorliegen müssen, waren absolute Höhenkarten nicht von Interesse.

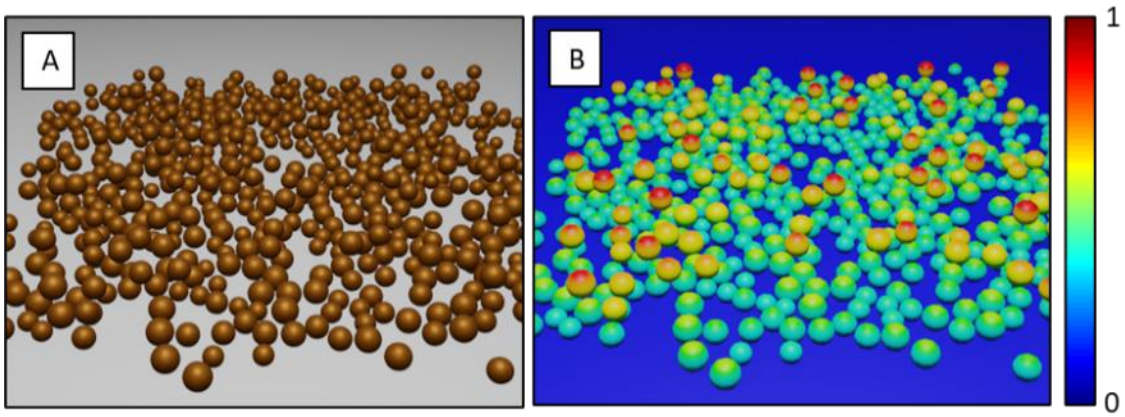


***Abbildung 40**: Relief der Szenerie von mehreren überlappenden Nanopartikeln nach dem Fall auf einen Probenträger. Für diese Simulation wurden 600 unterschiedlich große Kugeln verwendet.*

4.3.1 REM-Bildsynthese mittels CycleGAN

Die Synthese von künstlichen REM-Bildern aus simulierten Höhenkarten führt zu semirealistischen Aufnahmen diverser Nanopartikelformen. **Abbildung 42** verdeutlicht die Ergebnisse. Die synthetischen REM-Aufnahmen weisen ein realistisches Aussehen auf. Die Verteilung der Graustufen korreliert mit der Höhe der Partikel wie in **Abbildung 42-A1** und **A2** zu sehen. Auch können diverse Partikelformen synthetisiert werden. Die KI ist dabei in der Lage, REM-Bilder mit Partikelformen zu generieren, die nicht im Trainingsdatensatz der echten REM-Bilder enthalten waren. Dazu zählen z. B. pentagonale Bipyramiden (**Abbildung 42-B**) und Scheiben (**Abbildung 42-D**).

Der Realismus schließt zudem die Generierung eines Hintergrundes ein. In REM-Bildern sind teilweise verdeckte Partikel häufig als grau melierter Schatten abgebildet. Auch ist der Übergang von sichtbaren Partikeln zu „unsichtbar“ ein kontinuierlicher Prozess (**Abbildung 41**).

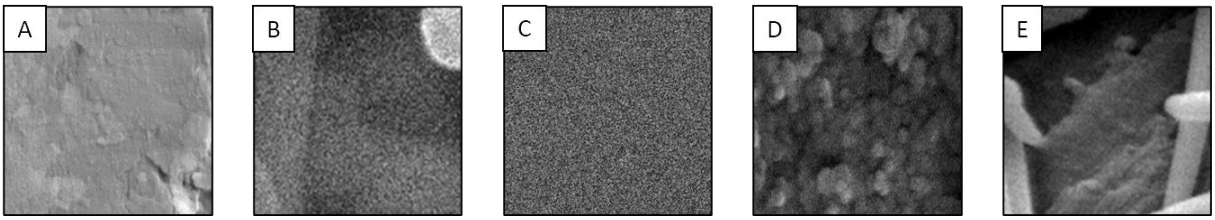


***Abbildung 41**: Typische Erscheinung von Hintergrund / Probenträger in SE-Bildern. Der Hintergrund in realen REM-Bildern unter anderen aus unspezifischer Variation des Untergrundes bestehen (**A, D, E**). Diese sind wahrscheinlich im Cu-Klebeband oder in Höhenunterschieden des Probenträgers zu begründet. Auch treten „unsichtbare Partikel“ auf. In Bildern, in denen die Partikel eine ausgeprägte Höhenvariation zeigen, sind teilweise verdeckte Partikel manchmal nur noch als Schattierungen angedeutet (**B**) oder sie verschwinden im Rauschen (**C**). Der Generator des GANs musste sich an verschiedene Formen von Hintergrund anpassen. Um dies zu erreichen, lernte dar Generator unspezifischen „Wölkchen“ Hintergrund zu generieren. Auf die Darstellung einer Skala wurde verzichtet. Die gezeigten Strukturen treten in allen Größenskalen auf.*

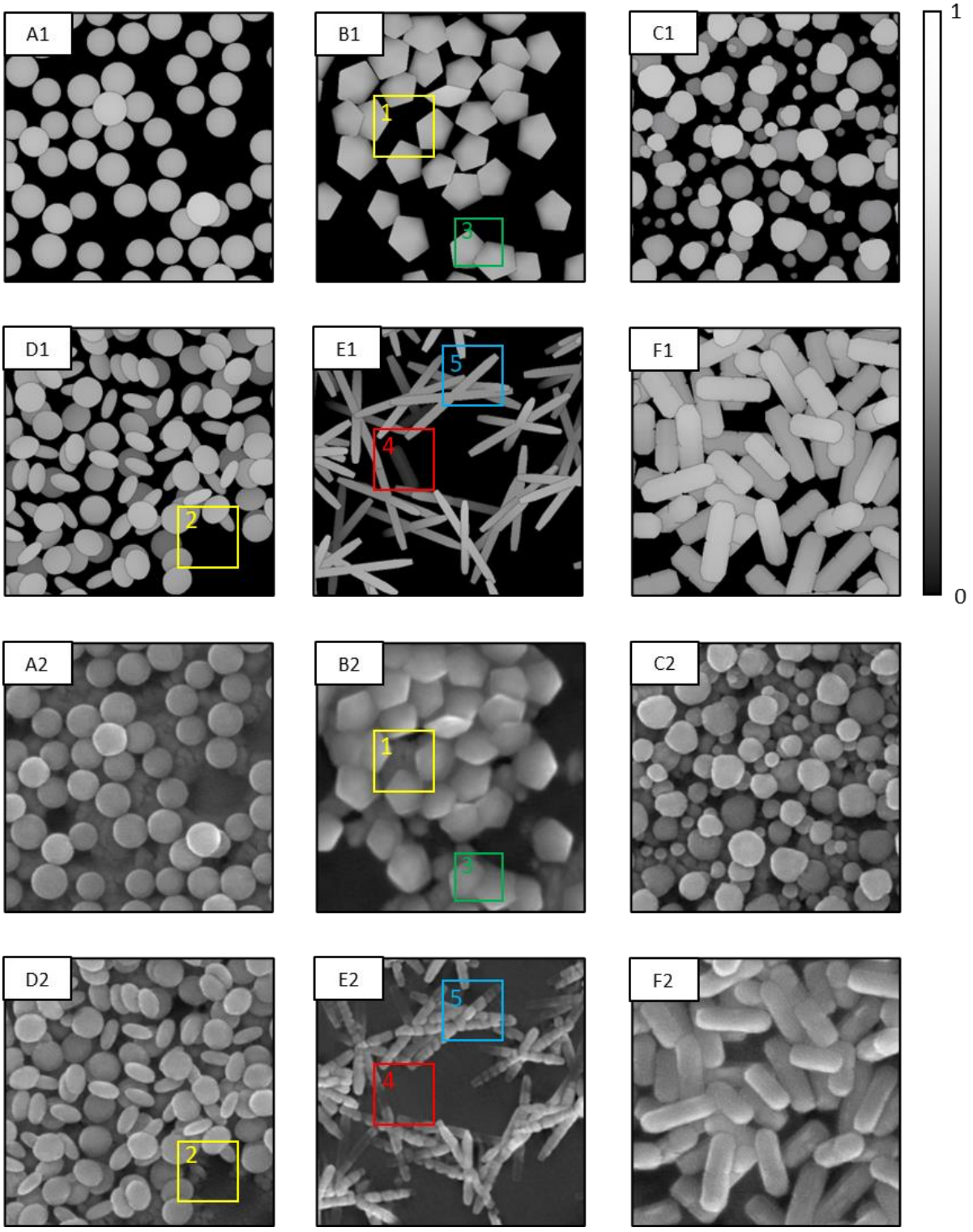


***Abbildung 42**: Darstellung verschiedener Nanopartikel im SE-Stil (**A2-F2**) aus simulierten Höhenkarten (**A1-F1**). Die Höhenkarten wurden aus Simulationen in Blender gewonnen und auf das Intervall [0-1] normalisiert. Null entspricht Hintergrund / Probenträger und 1 der höchsten Erhebung der Partikel. Vor der eigentlichen Umwandlung in ein REM-Bild fügt der Generator automatisch Gauß'schem Rauschen zu jedem Pixel hinzu. Dies gibt den Partikeln eine variable Oberflächenstruktur und erhöht den Realismus der Bilder. Auch der Hintergrund*

*profitiert durch Rauschen, da der Generator gezwungen wird, Hintergrundbereiche mit Textur zu füllen, ohne jedoch neue Partikel zu erschaffen (**A2**). Die funktioniert in den meisten Fällen, ist jedoch defacto nicht quantifizierbar. Die farbigen Boxen Bildern zeigen Schwachpunkte der KI auf.*

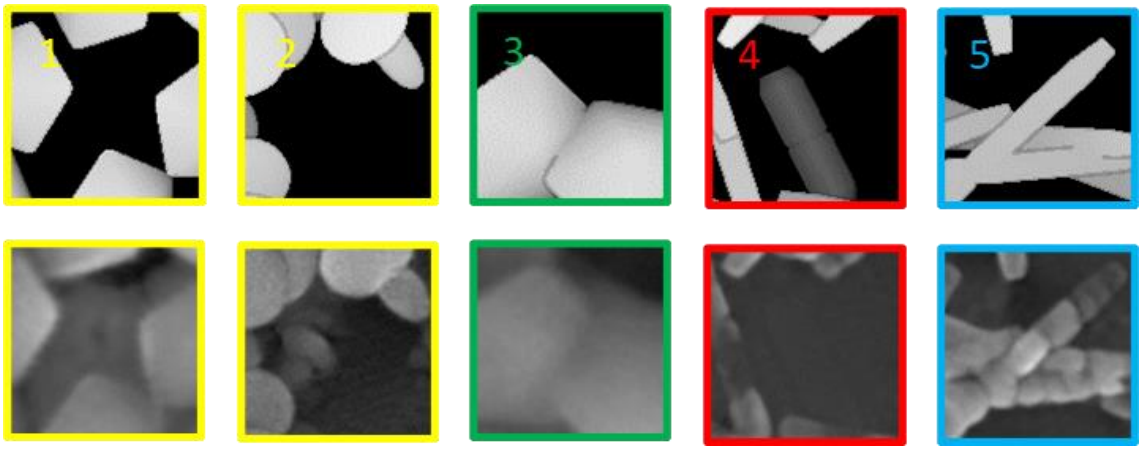


***Abbildung 43**: Schwachpunkte des GANs werden in den erzeugten REM-Bildern ersichtlich. Das GAN fügt als Hintergrund häufig "Wölkchen" aus undefinierbaren grauen Schlieren ein (**1-2**). Partikel verschmelzen in manchen Bereichen zu einem Partikel (**3**). In den Höhenkarten abgebildete Partikel verschwinden (**4**). Stäbchen werden in einzelne Kugeln aufgeteilt.*

Der Generator kann jedoch nicht selektiv Eigenschaften erlernen und diese nach Belieben abrufen, wie zum Beispiel diverse Variationen des Hintergrundes / Probenträgers. Aus den typischen Erscheinungen wie Übergänge von Partikeln in den Hintergrund unebener Probenträger wie Kohlenstoffklebeband oder ein fleckiger Hintergrund hat der Generator gelernt, Hintergrund nicht eben zu generieren. Der Realismus zwingt den Generator, Hintergrund zu erschaffen, die Funktionsweise des neuronalen Netzes zwingt den Generator zu Verallgemeinern. Der Mittelweg des Generators besteht in der Generierung von „Wölkchen“. Eine unebene und melierte Graustufen-Verteilungen wie sie in **Abbildung 42-B1** (gelber Kasten) sichtbar sind. Ebenfalls kann der Generator keinen variablen Probenträger erschaffen. Sind in Bereichen der Höhenkarte keine Partikel abgebildet, wird der Generator diese Bereiche mit dem typischen Auftreten eines REM-Probenträgers füllen. Dieser künstliche Hintergrund ist jedoch nicht so variabel bzw. realistisch wie ein realer Probenträger.

Eine weitere Problematik in der Erschaffung der Bilder liegt in der geringen Variabilität der Höhenkarten bezüglich der diversen Oberflächenstrukturen, die ein Nanopartikel annehmen kann. Die Oberflächenstruktur von SIO_2 Nanopartikeln unterscheidet sich von

Calciumphosphat oder TiO_2. Diese Oberflächenstrukturen sind zudem von der Größe der Partikel und der Vergrößerungsstufe des REMs abhängig. Auch hier muss der Generator des GANs lernen zu verallgemeinern. Der Generator wird die Schnittmenge der sichtbaren Oberflächenstrukturen des REM-Datensatzes erlernen und auf Höhenkarten übertragen. Um die Oberfläche von Objekten zu modifizieren bzw. zu diversifizieren und die Nullwerte des Hintergrundes zu eliminieren, wurde Gaußsches Rauschen eingesetzt. Das Rauschen führt zum Erlernen einer begrenzten Variabilität. Dennoch sind die erzeugten REM-Bilder sehr an Struktur verarmt, wie in **Abbildung 42** zu erkennen. Es wurden Versuche durchgeführt, den Generator an diverse Oberflächenstrukturen zu heranzuführen. Hierfür wurde Rauschen verwendet (**Abbildung 44**). Dazu gehören Perlin-Rauschen, weißes Rauschen, Salz- und Pfeffer-Rauschen und die grafische Modifizierung der Höhenkarten mittels Texturen aus Bildern. Strukturen wie Kratzer wurden dabei selektiv auf den Hintergrund angewendet, um diesen realistischer erscheinen zu lassen (**Abbildung 45**). Die Applikation von Rauschen oder die Modifizierung der Hintergrundstruktur führte zu keiner Verbesserung bezüglich des Realismus in den erzeugten Bildern. Es sind zwar begrenzte Variationen sichtbar, aber nicht in der Vielfalt, wieVielfalt, wie sie in realen REM-Aufnahmen vorkommen. Nicht mal in der Vielfalt wie sie im REM Datensatz des Generatortrainings vorkam.

Positiv hingegen ist die Leistung des Generators hinsichtlich des Fresnel-Effektes zu erwähnen. Kanten der Partikel zeigen oftmals die typische Aufhellung. Da dieser Effekt unabhängig von Form oder Struktur der Partikel ist, konnte der Generator diese Eigenschaft gut erlernen.

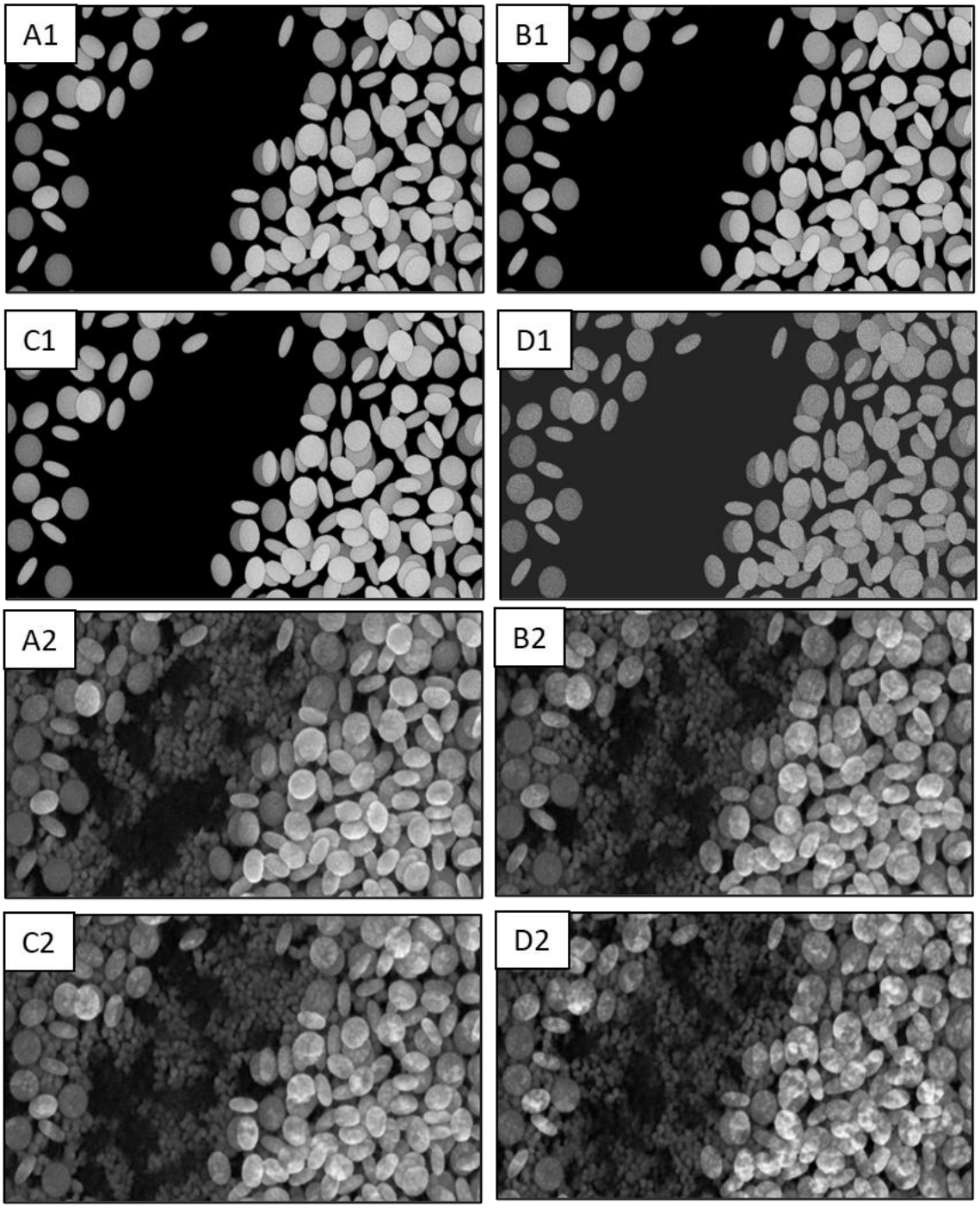


***Abbildung 44**: Modifizierung der Partikeloberfläche durch Gauß'sches Rausches in den Höhenkarten (**A1-D1**). Das Rauschen wurde selektiv auf die Bildbereiche angewendet, in denen Partikel liegen. Durch die Veränderung wurde auch der Hintergrund um einige Intensitätsstufen heller. Das Rauschen besitzt in allen Bildern einen Mittelwert von 0 und eine Standardabweichung von 2,55 (**A1**), 12,75 (**B1**), 25,5 (C1), und 38,25 (**D1**) Intensitätsstufen. Die Partikel erscheinen in den erzeugten synthetischen REM-Bildern nicht rauer, sondern besitzen mit zunehmendem Rauschen eine wellige Struktur. Einzig das Rauschen, wie in **A1** gezeigt, resultiert in REM-Bildern mit annehmbarem Realismus.*

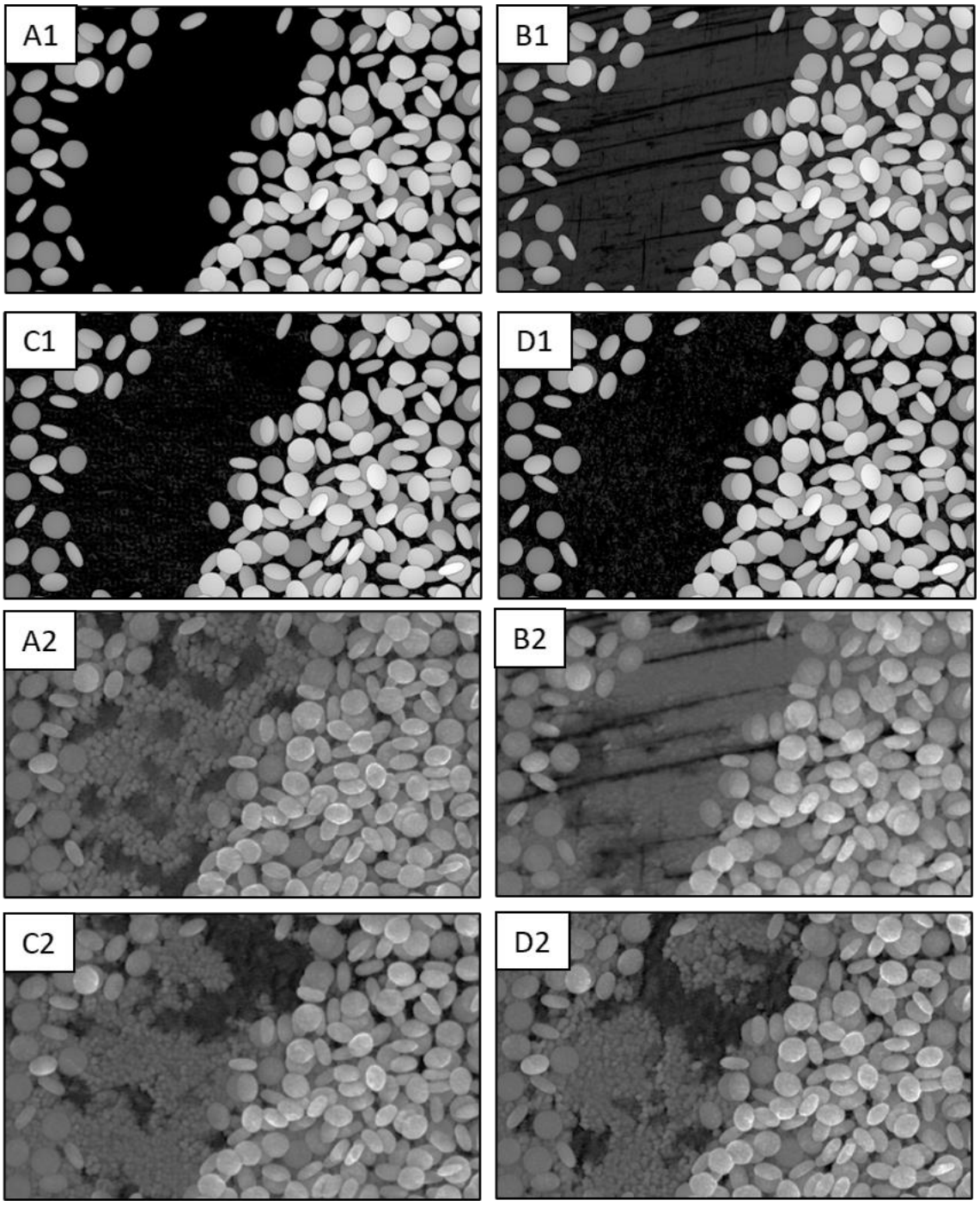


***Abbildung 45**: Um Kratzer oder andere Variationen des Probenträgers darzustellen, wurden Texturen verwendet, um selektiv Bildbereiche zu verändern. [86] Der Generator berücksichtigt die Texturen, um den Probenträger variabler zu gestalten. Dennoch wird der Probenträger weitgehend zu dem allgemeinen „Wölkchen" Hintergrund modifiziert (**B2-D2**), wie er auch in Bildern ohne Modifizierung des Hintergrundes angewendet wird (**A1**). Die genutzten Texturen halfen nicht, dass der Generator den Hintergrund realistischer gestaltet.*

Wie in **Abbildung 42-E2** zu erkennen ist, wurden lange Stäbchen vom Generator in einzelne Kugeln aufgespalten. Dies liegt vermutlich an der Architektur des Netzwerkes bzw. der Größe der Eingabeschicht des Generators wie auch des Diskriminators. Die Eingabeschicht besaß in beiden Netzwerken eine Größe von 512x512 Pixeln. Das Training mit größeren Bildern war technisch nicht realisierbar. Der Diskriminator erlernte in mehreren Faltungs-Schichten, die Eigenschaften echter REM-Bilder zu identifizieren und die Realismus-Bewertung kodiert auszugeben. Die Realismus-Bewertung ist eine Matrix. Bedingt durch die wiederholte Anwendung von Faltungen ist sie um den Faktor 8 kleiner als das Bild (**Abbildung 46**). Das rezeptive Feld des Diskriminators entspricht somit einem Bereich von 64x64 Pixeln. Werte von ≥ 1 bedeuten der Diskriminator hält den Bildbereich für echt. Werte von ≤ 0 bedeuten der Diskriminator schätzt diesen Bildbereich als künstlich ein. Werte nahe 0,5 sprechen für eine Unsicherheit des Diskriminators. Der Mittelwert der Matrix ist die Gesamtschätzung des Bildes bzgl. seines Realismus. Zwangsläufig werden mit dieser Methode Partikel von dem rezeptiven Feld abgeschnitten.

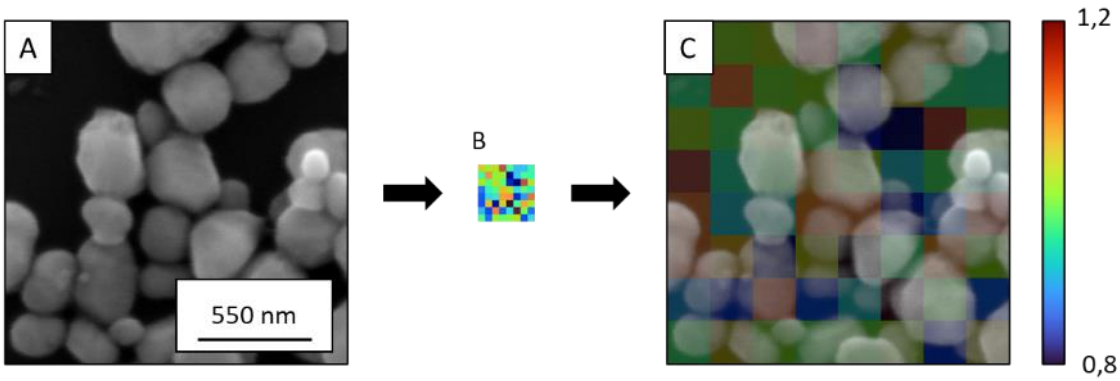


***Abbildung 46**: Einfluss des rezeptiven Feldes auf Einzelwerte der Bewertungsmatrix des Diskriminators. Ein 512x512 Pixel großes Bild, in diesem Fall ein reales REM-Bild (**A**), wird vom Diskriminator auf seinen Realismus hin untersucht. Das Ergebnis ist eine Matrix (**B**), deren Komponenten jeweils ein rezeptives Feld von 64x64 Pixel aufweisen. Die Matrix ist dementsprechend 8x8 Pixel groß (512 px / 64 px = 8 px). Vergrößert man die Matrix auf 512x512 Pixel, zeigt diese an wie real der Diskriminator Bereiche des Eingangsbildes bewertet hat (**C**). Ein größeres rezeptives Feld, beispielsweise durch das Hinzufügen einer weiteren Schicht könnte dem Diskriminator helfen, größer-skalierte Partikel korrekt einzuschätzen. In diesem Fall liegt die mittlere Bewertung des Bildes bei ca. 0,95. Der Diskriminator befindet das Bild als real.*

Große Partikel werden von mehreren rezeptiven Feldern abgeschnitten. Hierdurch wird ein großes Partikel nicht vollständig bewertet werden können, was zu einer geringen Einschätzung des Diskriminators führt. Schlussendlich zwingt der Generator große Partikel wie z. B. lange Stäbchen in eine Form, die der Diskriminator als real einstuft. Es entstehen Ketten aus Kugeln, wie in **Abbildung 42-E2** zu sehen sind. Ein möglicher Ansatz zur Minimierung dieser Fehlerquelle wäre ein größeres neuronales Netz zu verwenden. Eine Vertiefung des Diskriminators um eine weitere Faltungs-Schicht würde das rezeptive Feld auf 128x128 Pixel erhöhen, zwei weitere Schichten auf dementsprechend 256x256 Pixel. Eine weitere Möglichkeit bestünde in der Vergrößerung der Eingangsbilder. Dadurch könnten große Partikel, wie Stäbchen vollständig abgebildet werden. Aus technischen Gründen waren weder die Vertiefung des Netzwerkes noch die Vergrößerung der Eingangsbilder möglich.

In der Trainingsroutine nehmen die Kosten des Netzwerkes künstlich ab Epoche 80 ab, um die Kosten, welche sich aus den Identitätskosten (L_i) und den zyklischen Kosten (L_c) ergeben, gegen Null gehen zu lassen. Dies ist nötig, um einen Punkt zu erreichen, an dem das GAN nur noch minimale Änderungen seiner Parameter vornimmt. Dies bedeutet nicht, dass der Generator perfekt ist. Jedoch kann der Generator ab einem gewissen Punkt keine Eigenschaften des Datensatzes mehr erlernen, da der Diskriminator alle Bilder des Generators als reale REM-Bilder klassifiziert und L_i und L_c schwanken. Um eine Einschätzung über die Qualität der Bilder zu erhalten, wurde eine Umfrage unter chemisch erfahrenen Personen vorgenommen. 16 künstliche und 12 reale REM-Bilder wurden zur Klassifizierung gezeigt. 8 künstliche Bilder wurden als real eingestuft, während 2 reale Bilder als künstlich eingestuft wurden.

Es kann zusammengefasst werden, dass das GAN noch viele Schwachpunkte aufweist. Der Realismus der Bilder ist nach menschlicher Einschätzung nicht gegeben. Es ist zudem nicht in der Lage, eine ausreichende Variabilität von Hintergründen oder Partikeltexturen zu erzeugen. Oft werden Partikel, die auf den Höhenkarten abgebildet sind, nicht in den synthetischen REM-Bildern abgebildet. Die Spaltung oder das Verschmelzen von Partikeln konnte nicht angegangen werden. Eine Erweiterung der Netzwerke könnte hier eine Verbesserung erbringen. Technisch war dies limitiert.

4.3.2 UNet++ trainiert mittels künstlicher SE-Aufnahmen

Die Leistung des Modells zur Segmentierung von SE-Aufnahmen ist in **Tabelle 6** dargestellt. Nachdem im Training des UNet++ Modells auf reale Daten (*Modell R; reale Trainingsdaten*) bereits eine höhere IoU bei der Nutzung von distanzbasierten Gewichtungskarten festgestellt wurde, wurde das hier trainierte Modell (*Modell S; synthetische Trainingsdaten*) ausschließlich auf distanzbasierte Gewichtungskarten trainiert. Die Gewichtungskarten wurden, wie bereits für den realen Datensatz aus den Annotationsmasken berechnet.

***Tabelle 6**: Ergebnisse der Validierung des UNet++ Modells trainiert auf synthetische SE-Bilder und künstliche Annotationsmasken aus simulierten Partikeln. Für einen Vergleich ist die Leistung des Modell R ebenfalls abgebildet. Modell R wurde ebenfalls auf distanzbasierten Gewichtungskarten trainiert.*

	IoU	Precision	Recall	Rand Index
UNet++ *Reale Daten*	93,20 %	97,39 %	95,61 %	2,34 %
UNet++ *Künstliche Daten*	80,52 %	91,78 %	87,06 %	8,07 %

Die Leistung des Modells S ist deutlich geringer als die des Modell R. Sämtliche Metriken zeigen denselben Trend. Dieser Trend mag mehrere Gründe haben. Ein Grund, warum künstliche Daten den Lernprozess behindern können, besteht darin, dass sie die realen REM-Bilder, für die das Modell verwendet werden, möglicherweise nicht genau darstellen. Im Falle der Segmentierung können die Bilder, die für das Training genutzt werden, einen zu geringen Realismus aufweisen. Das Modell kann dann möglicherweise nicht gut auf die realen Bilder verallgemeinern und bei der Anwendung eine schlechte Leistung erbringen. Die synthetischen SE-Bilder unterscheiden sich in mehrfacher Hinsicht von realen Bildern. Der wesentliche Unterschied besteht darin, dass die synthetischen Bilder mithilfe von Computeralgorithmen generiert wurden und nicht durch die physikalischen Gesetze der realen Welt eingeschränkt werden. Die genutzten Simulationen, um den Fall der Partikel nachzuahmen, beinhalten beispielsweise keine Effekte, die mit Nanomaterialien assoziiert werden. So werden Agglomeration von Partikeln und Trocknungseffekte auf dem Probenträger ignoriert. In einer realen nasschemischen Synthese von Nanopartikeln können eine Vielzahl von Nebenprodukten entstehen oder die Synthese schlägt vollends fehl. Die entstandenen

Produkte spiegeln daher eine große bandbreite möglicher Effekte wider. Auch kennt das GAN um die SE-Bilder zu erzeugen, nicht den Unterschied zwischen den im Datensatz enthaltenen Materialien, Störungen der REM-Bilder oder physikalischen Effekten. Infolgedessen können synthetische Bilder, Elemente und Effekte enthalten, die in einem realen Bild unmöglich zu erfassen wären, wie z. B. surreal einheitliche und glatte Objekte oder bestimmte Partikelanordnungen, die untypisch für reale Nanopartikel wären. Die Umfrage bezüglich des Realismus der Bilder stützt diese Vermutung. 50 % der künstlichen Bilder wurden von einer menschlichen Expertengruppe korrekt identifiziert. Korrekt identifiziert bedeutet in diesem Falle, dass der menschliche Analyst die Bilder als synthetisch erkannt hat. Dies legt nahe, dass nur ca. 50 % der Bilder den typischen REM-Look aufweisen. Jedoch ist die Umfrage mit weniger als 30 Personen durchgeführt worden. Zudem wurden nur 16 künstliche Bilder verwendet. Die Umfrage ist daher wenig repräsentativ. Betrachtet man die gesamte Bandbreite produzierter synthetische Bilder, wird die Anzahl der vom Menschen als künstlich identifizierten Bilder sicherlich steigen. Dies ist aber nur eine Vermutung und würde weitere Daten benötigen.

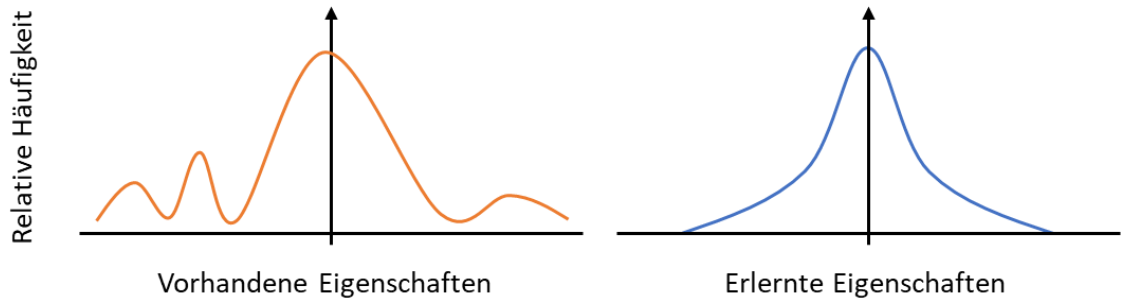


***Abbildung 47**: Schematische Darstellung der Bild-Eigenschaften eines beliebigen Datensatzes gegenüber den erlernten Eigenschaften. [25] Es ist ersichtlich, dass innerhalb eines Datensatzes einige Eigenschaften häufiger auftreten, während andere schwächer ausgeprägt sind. Wird ein GAN auf diese Eigenschaften trainiert, lernt der Generator des GANs daraus zu generalisieren. Die vom Generator erzeugten Bilder weisen im Vergleich zu realen Daten einen eingeschränkten Eigenschaftsraum auf.*

Ein weiterer Grund, warum künstliche Daten den Lernprozess behindern können, ist, dass sie möglicherweise nicht genügend Variation der Oberfläche oder Vielfalt der Partikelformen enthalten. Wenn ein KI-Modell auf künstlich erzeugten Daten trainiert wird, die zu einheitlich sind oder denen es an Diversität mangelt, ist es möglicherweise einer nicht ausreichend großen Auswahl an Beispielen ausgesetzt, um effektiv zu lernen. Dies kann zu einem übermäßig

vereinfachten oder verzerrten Modell führen, das bei realen Daten keine hohe Leistung erbringt.

Abbildung 48 visualisiert die Ähnlichkeit der erzeugten künstlichen Daten und der realen REM-Bilder des Datensatzes. Hierfür wurden aus den künstlichen wie auch realen Bildern ca. 800 zufällige quadratische Patches mit einer Kantenlänge von 128x 128 Pixeln ausgeschnitten. Anschließend wurden die Eigenschafts-Tensoren mittels eines vortrainierten neuronalen Netzes berechnet. Als vortrainierten Eigenschafts-Extraktor wurde VGG19 genutzt. [18] VGG19 ist ein kompaktes, leistungsstarkes neuronales Netz. Dies ist auf den ImageNet-Datensatz vortrainiert worden. [84] Dadurch konnte ein menschlicher Bias ausgeschlossen werden. Anschließend wurden die Patches gemäß ihren berechneten Eigenschaften nach der Methode von van der Maaten visualisiert. [85] Hierüber ist es möglich, die Eigenschaften der realen wie auch künstlichen Bilder miteinander zu vergleichen. Je näher das neuronale Netz die Eigenschaften von zwei Bildern befindet, desto näher wurden diese zueinander geplottet. Wären reale wie auch künstliche Bilder identisch, können diese nicht voneinander unterschieden werden. Das Resultat der Visualisierung wären durchmischte Bilder, in denen die Patches nicht in „real“ oder „künstlich“ geclustert werden können. Dies ist hier nicht der Fall. Künstliche Patches (lila umrandet) liegen nahe beieinander und Grenzen sich lokal von den realen Patches (grün umrandet) ab. Zudem liegen künstliche Patches in einem engen Bereich (lila Ellipsen) während reale Bilder weitreichend und divers vorliegen. Dies deutet auf eine geringe Variabilität der künstlichen Bilder.

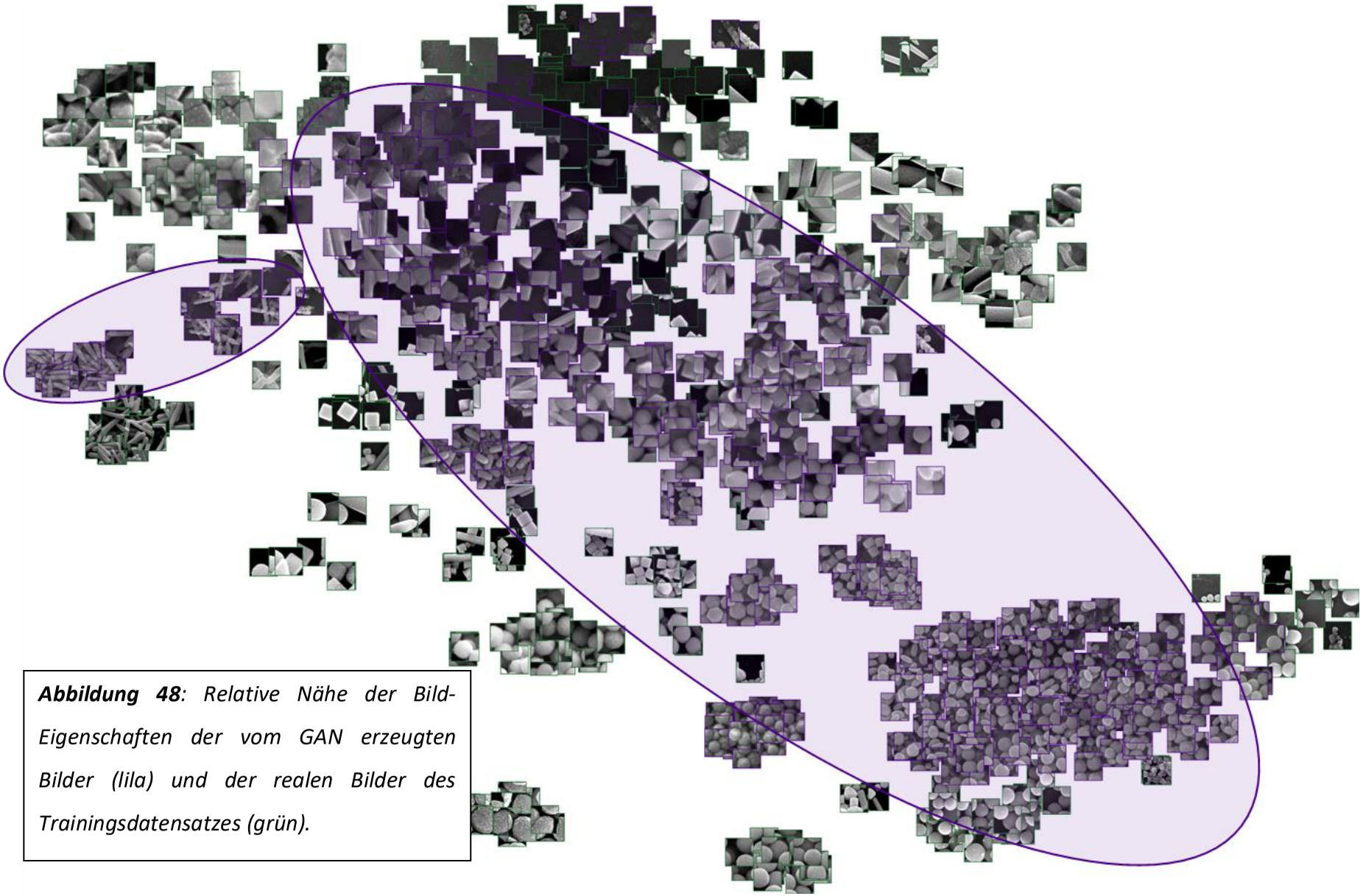


***Abbildung 48**: Relative Nähe der Bild-Eigenschaften der vom GAN erzeugten Bilder (lila) und der realen Bilder des Trainingsdatensatzes (grün).*

***Abbildung 48 – Fortsetzung**: Gezeigt sind zufällige Bildausschnitt der Bilder des realen GAN-Trainingsdatensatzes und der synthetischen Bilder. Je näher die Bildeigenschaften zweier Ausschnitte sind, desto ähnlicher sehen diese aus. Dementsprechend werden sie näher zueinander geplottet. Aus der Abbildung geht hervor, dass die künstlichen REM-Bilder alle sehr ähnlich sehen. Sie liegen in einem relativ engen Bereich zueinander (lila Ellipse). Dagegen sind die Bildausschnitte realer REM-Bilder divers und liegen gestreut vor. Dies zeigt die bereits in **Abbildung 47** vermutete Verteilung von Bildeigenschaften. Das dahin gehende Fehlen an Variation in den erzeugten Bildern hindert des UNet++ Model am Erlernen eben dieser Eigenschaften. Die Abbildung wurde nach der Visualisierungsmethode von van der Maaten [85] mit den Standardeinstellungen von* sk-Learn *erstellt.* [87]

Durch die bisherige Betrachtung der künstlichen Bilder konnte festgestellt werden, dass diese weder für den menschlichen Betrachter noch für die KI real aussehen. Die Variabilität der erzeugten Bilder ist gering und die generierten Bildeigenschaften liegen sehr eng zusammen. Hieraus ergibt sich die geringe Fähigkeit des Segmentierungsmodels zur Adaptierung an variable reale Daten. Dieser Trend ist in der Literatur bereits bekannt, auch wenn die hier gezeigte Interpretation der Daten bereits eine deutlich bessere Überlagerung von Bildeigenschaften zeigt als bisher. [61] Als mögliche Lösung könnte das Netzwerk tiefer gemacht werden. Zusätzliche Schichten und somit zusätzliche Parameter könnten dem Generator die Möglichkeit geben, sich adaptiver zu verhalten. Weiterhin könnte die Visualisierungsmethode selbst als Verlustfunktion genutzt werden, wodurch das Netzwerk nochmals verstärkt lernen könnte, die Bilder ähnlicher an reale REM-Aufnahmen anzupassen. So könnte beispielsweise versucht werden, den Abstand der bisher gut separierbaren Cluster zu minimieren. Dieser Versuchsaufbau ist jedoch eine komplexe Idee, die in dieser Arbeit nicht weiter verfolgt wurde.

Das Ziel der Erstellung künstlicher Bilder bleibt es, einen Datensatz zu erstellen, der es einem Segmentierungsmodell ermöglicht, Partikel in SE-Bildern genau zu identifizieren und dementsprechend die Partikelgrößen zu identifizieren. Für die Validierung dieses Ziels wurde die Größe der Nanopartikel in den Annotationen des SE-Validierungsdatensatzes mit der Größe der Partikel aus den Segmentierungen von Model S verglichen. **Abbildung 49** zeigt, wie bereits **Abbildung 33**, den die Partikelgrößenverteilung aller Partikel des Validierungsdatensatzes und die Auswertung der mittels Modell S erhaltenen Segmentierungen. Dies erlaubte eine erste Abschätzung, ob das Modell S für die Auswertung

von SE-Bildern geeignet war. Die Ergebnisse in **Abbildung 49** zeigen dabei auf, dass die Größen der Annotationen und der Segmentierungen stark überlappen.

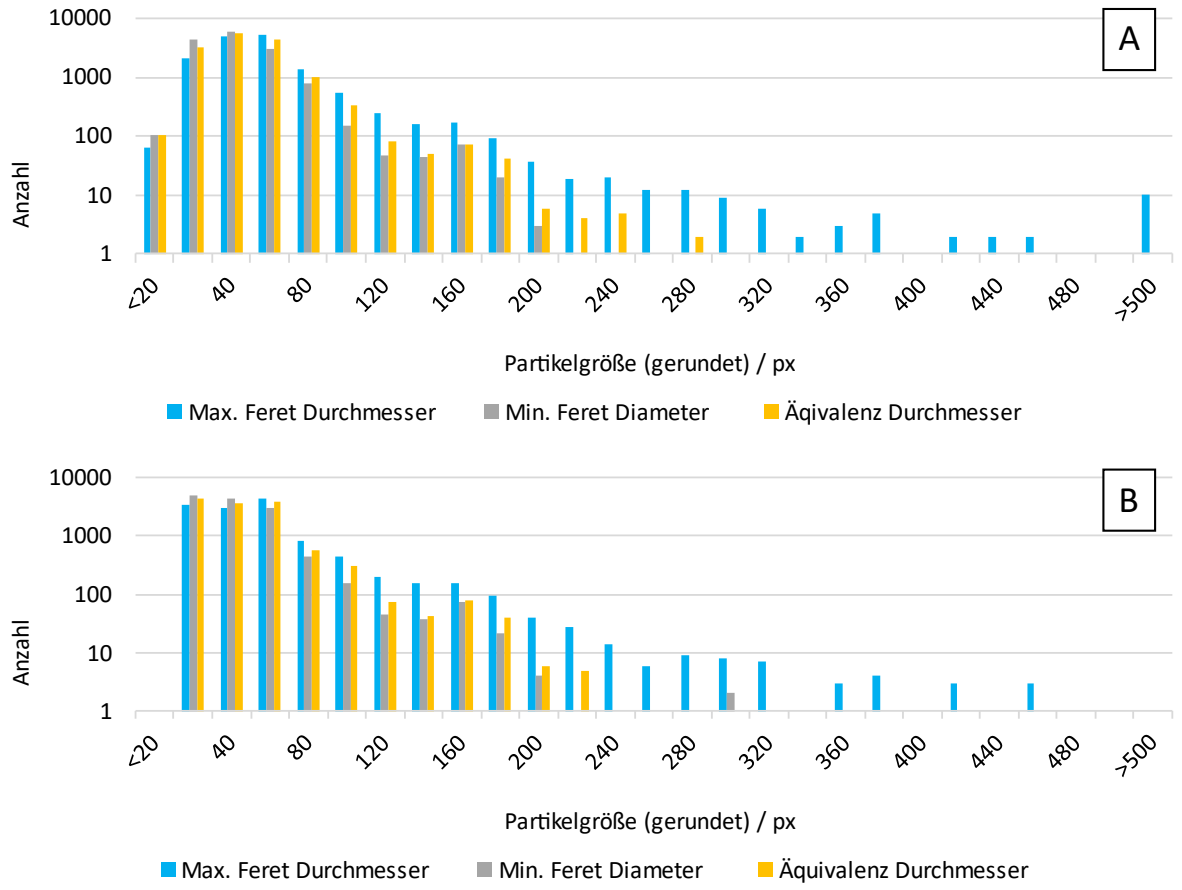


***Abbildung 49**: Auswertung aller im SE-Validierungsdatensatz enthaltenden Partikel. Die Grafik zeigt die Verteilung der Partikelgrößen in den händisch erstellten Annotationen (**A**) gegenüber den Partikelgrößen der Segmentierungen des* Modell S *(**B**). Die gezeigte Partikelgrößenverteilung in **A** ist identisch zu der gezeigten in **Abbildung 33-A**. Die Partikel sind nicht nach ihrer Form klassifiziert. In den Annotationen wurden insgesamt ca. 15.000 einzelne Partikel markiert. Das Modell war in der Lage, ca. 13.000 Partikel zu lokalisieren. Dies entspricht der Partikelanzahl, die auch vom realen SE-Model identifiziert wurde.*

4.4 Einzelauswertungen von typischen REM-Bildern

Die bisher gezeigten Ergebnisse sind spezifisch für das jeweilige KI-System. Für die folgenden Analysen wurden die einzelnen KI-Systeme zusammengefügt, um eine nahtlose Analyse vollständiger REM-Bilder im SE- oder STEM-Modus durchführen zu können.

Der vollständige Workflow für die Analyse von REM-Bildern unabhängig ihrer spezifischen Art (STEM, SE) ist in **Abbildung 50** gezeigt. Um vollständige Bilder auszuwerten, wurde ein Bild zunächst mit dem dafür trainierten Segmentierungs-Modell in ein binäres Bild segmentiert. Die Position, Höhe, Breite und Fläche der einzelnen Vordergrundobjekte wurden direkt aus der Segmentierung abgeleitet. Anhand der Position der Objekte in der binären Segmentierung sowie der Höhe bzw. Breite der Partikel konnten Bereiche, die nach Einschätzung des Segmentierungsmodells ein Partikel enthalten, aus dem REM-Bild ausgeschnitten werden. Dabei war es zunächst nicht von Belangen, ob die Partikel vollständig sichtbar sind. Teilbereiche von Partikeln oder Agglomerate wurden genauso ausgeschnitten wie fehlerhaft segmentierter Hintergrund. Jeder ausgeschnittene Patch wurde skaliert, sodass die lange Seite des Patches die benötigte Breite aufweist, mit welcher das jeweilige Klassifizierungsmodell trainiert wurde. Um ein quadratisches Bild zu erhalten, wurden weitere Pixel mit Nullwerten aufgefüllt. Anschließend wurden die Patches klassifiziert. STEM-Partikel wurden als „Hintergrund“, „Agglomerat“ oder entsprechend ihrer Morphologie eingeteilt. SE-Partikel in „Hintergrund“ oder ihre entsprechende Morphologie. Verdeckte Partikel wurden der Klasse „Hintergrund“ zugeordnet. Dies hatte den Vorteil, dass im Training der Klassifizierungs-Modelle nur die Klasse „Hintergrund“ trainiert werden musste, um falsch segmentierte Bereiche und verdeckte Partikel zu erfassen, anstatt zwei getrennte Klassen („Hintergrund“ & „Verdeckt“). Verdeckte Partikel und falsch segmentierte Bereiche wurden in der weiteren Analyse der Partikelgröße nicht berücksichtigt. Agglomerate in STEM-Bildern hingegen wurden weiter analysiert, um eine mittlere Anzahl an Partikeln zu berechnen, die einem Agglomerat angehören. Die selektive Berechnung der Größe einzelner Formen wird erst durch die Einteilung eines Klassifizierungsmodells möglich gemacht. Insofern ist es ein engmaschiges Zusammenspiel aus Segmentierung, Klassifizierung und selektiver Partikelauszählung, die den gezeigten Workflow menschlich bzw. übermenschlich agieren lässt.

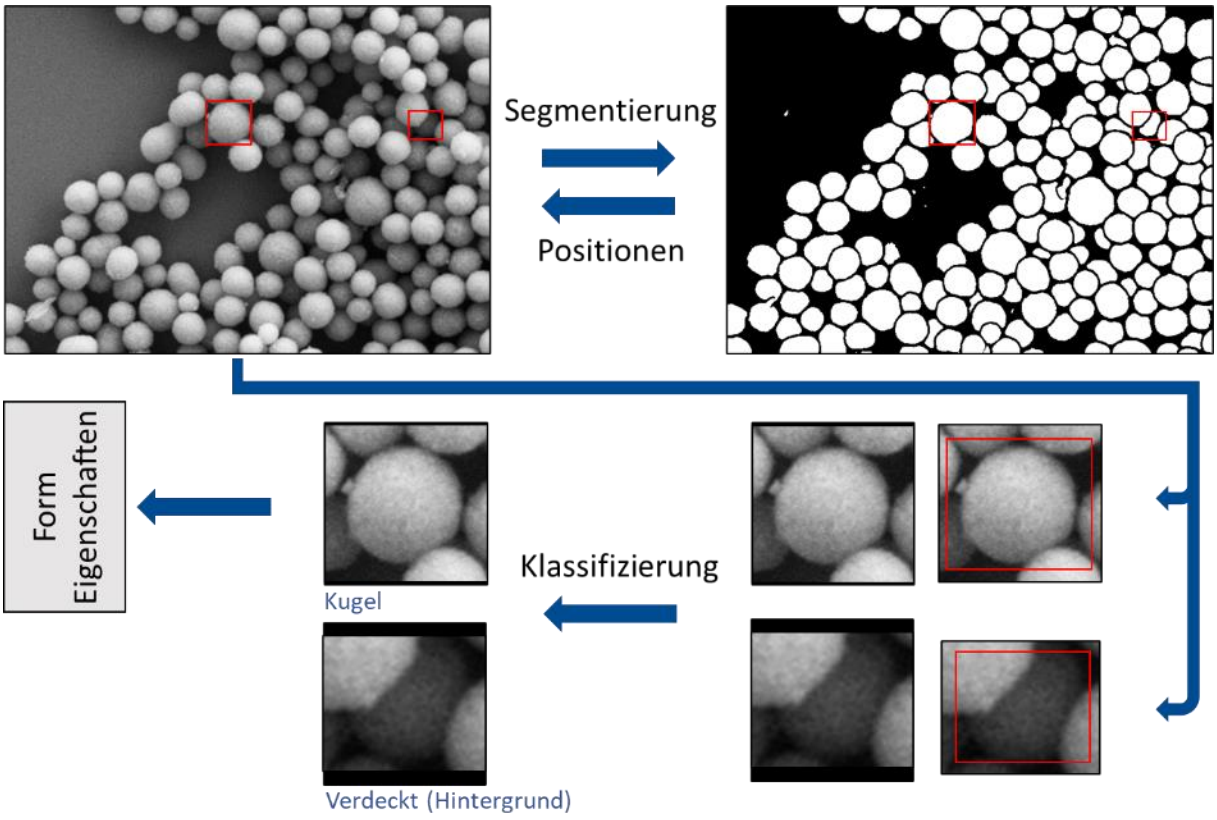


***Abbildung 50**: Workflow zur Segmentierung von REM-Bildern im SE- oder STEM-Modus. Nach der Segmentierung wurden einzelne Vordergrundbereiche als "Partikel" gekennzeichnet. Ihre Position wird gespeichert. Gemäß Position und Größe der einzelnen Partikel werden rechteckige Bildbereiche (rote Rechtecke) in den REM-Bildern ausgeschnitten und für die Klassifizierung vorbereitet. Die Größe des ausgeschnittenen Bereichs ist dabei minimal größer als die Partikel selbst. Die einzelnen Partikel werden in die bekannten Klassen eingeteilt. Nur Partikel, die nicht als „Verdeckt/Hintergrund" erkannt wurden, werden bezüglich ihrer morphologischen Eigenschaften weiter ausgewertet. Abbildung entnommen und verändert aus Bals et al.* [81]

Der gezeigte Workflow wurde für eine Reihe von SE- und STEM-Bildern durchgeführt, um Schwachstellen der KI zu detektieren, die in der Betrachtung der gesamten Datensätze nicht zu erfassen waren. Zudem wurden alle SE-Bilder mit 2 verschiedenen Modellen segmentiert. Dem Modell trainiert auf realen Daten und menschlichen Annotationen (Modell R, real) und dem Modell trainiert auf künstlichen Daten und simulierten Annotationen (Modell S, synthetisch). Die Klassifizierung erfolgte mittels der vorgestellten Modelle. Dies sollte zeigen, inwiefern sich künstliche Daten qualitativ für das Training der gezeigten Modelle eignen.

Zusätzlich zur Qualitätskontrolle der einzelnen Partikel bezüglich Überdeckung oder falsch Segmentierung wurde ein „Konfidenz-Limit“ in der Klassifizierung festgelegt. Betrug die maximale Wahrscheinlichkeit der Klassifizierung eines Partikels weniger als 75 % wurde dieses als „Unbekannt“ eingestuft. [81] Die KI war sich bezüglich der Morphologie dieses Partikels „unsicher“. Das Konfidenz-Limit entspricht der Wahrscheinlichkeit einer menschlichen Gruppe ein Partikel einer der im Training verwendeten Morphologien zuzuordnen. [69] War sich die KI bei einem Partikel unsicher, wurde dieses aus der Auswertung ausgeschlossen. Die Wahrscheinlichkeit, dass es sich bei einem solchen Partikel um ein falsch identifiziertes Partikel oder Hintergrund handelte, war zu groß. Erfahrungen im Umgang mit diesem System haben dies bestätigt.

Die Partikelgrößenverteilungen wurden für jede Probe erstellt und mit den Auswertungen der menschlichen Analysten verglichen. Für die unterschiedlichen Partikelformen wurden die typischen morphologischen Eigenschaften (Radius & Kantenlänge) aus bekannten Formeln abgeleitet. Dies erlaubte die Fläche direkt in die Kantenlänge (bzw. Radius für Kugeln und kugelartige Partikel) umzurechnen, anstatt wie bisher üblich den Feret Durchmesser der Partikel zu bestimmen. Somit konnten spezifische Größen für jede Form berechnet werden. Für die Auswertung von Kugeln, Kugelartigen und unbekannten Partikel wird im Folgenden der Äquivalenzradius genutzt. Um ein Partikel zu beschreiben. Für alle weiteren Formen wird die Kantenlänge gezeigt.

Die untersuchten Bilder stammen von verschiedenen Personen, wurden mittels unterschiedlicher Synthesen hergestellt und von den Herstellern bezüglich der Partikelgröße selbst ausgewertet. Während die bisherige Auswertung quantitativer Natur war, sollen die nun gezeigten Bilder qualitativ die Ergebnisse der Verknüpfung von Segmentierung und Klassifizierung zu verschiedenen Menschen zeigen. Die ausgezählten Partikel pro Bild lagen im Bereich on 50-100 Partikel. Im Normalfall wurden 100 Partikel der gewünschten Morphologie ausgezählt.

Zuletzt soll angemerkt werden, dass die gezeigten Bilder mittels des veröffentlichten Workflows angefertigt wurden. [81] Die Legende wurde von der Software automatisch generiert.

4.4.1 Ag Nanopartikel bei 100.000-facher Vergrößerung (STEM)

Die Auswertung von Ag Nanopartikeln ist in **Abbildung 52** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 51** & **Tabelle 7** gezeigt. Die Probe zeigt eine gemischte Population von Würfeln, Tetraedern und undefinierbaren Partikeln, die in der 2D Darstellung des STEM-Modus als Quadrate und Dreiecke erscheinen. Das Ziel der Synthese waren Partikel definierter Form. Es folgt die Auswertung von quadratischen und dreieckigen Partikeln.

***Tabelle 7**: Auswertung der Partikelgrößen aus **Abbildung 52** sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	**Ør / nm**	**n**
Mensch	Quadrate (Kantenlänge)	70 ± 21	40
	Dreiecke (Kantenlänge)	120 ± 17	20
Modell	Quadrate (Kantenlänge)	56 ± 25	62
	Dreiecke (Kantenlänge)	127 ± 19	22

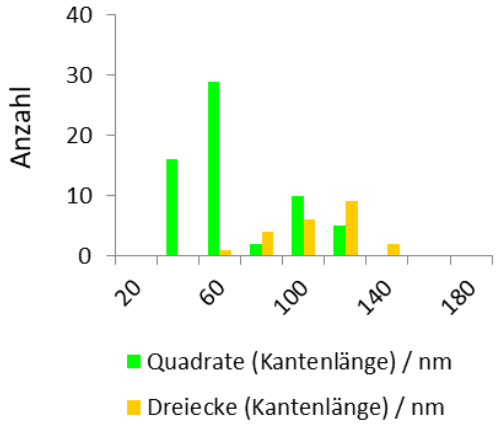


***Abbildung 51**: Partikelgrößenverteilung als Histogramme, welche in **Abbildung 52** bei 100.000-facher Vergrößerung gezeigt sind.*

Es zeigt sich, dass die Partikel mit ausreichender Präzision erkannt werden. Trotz der Verzerrung einzelner Partikel, wie zum Beispiel an den Sechsecken zu sehen ist, werden diese erkannt. Die Hauptform in diesem Bild sind Quadrate, welches ebenfalls stabil segmentiert und weiterhin klassifiziert werden können, sodass eine sichere Auswertung der mittleren Größe erfolgen kann.

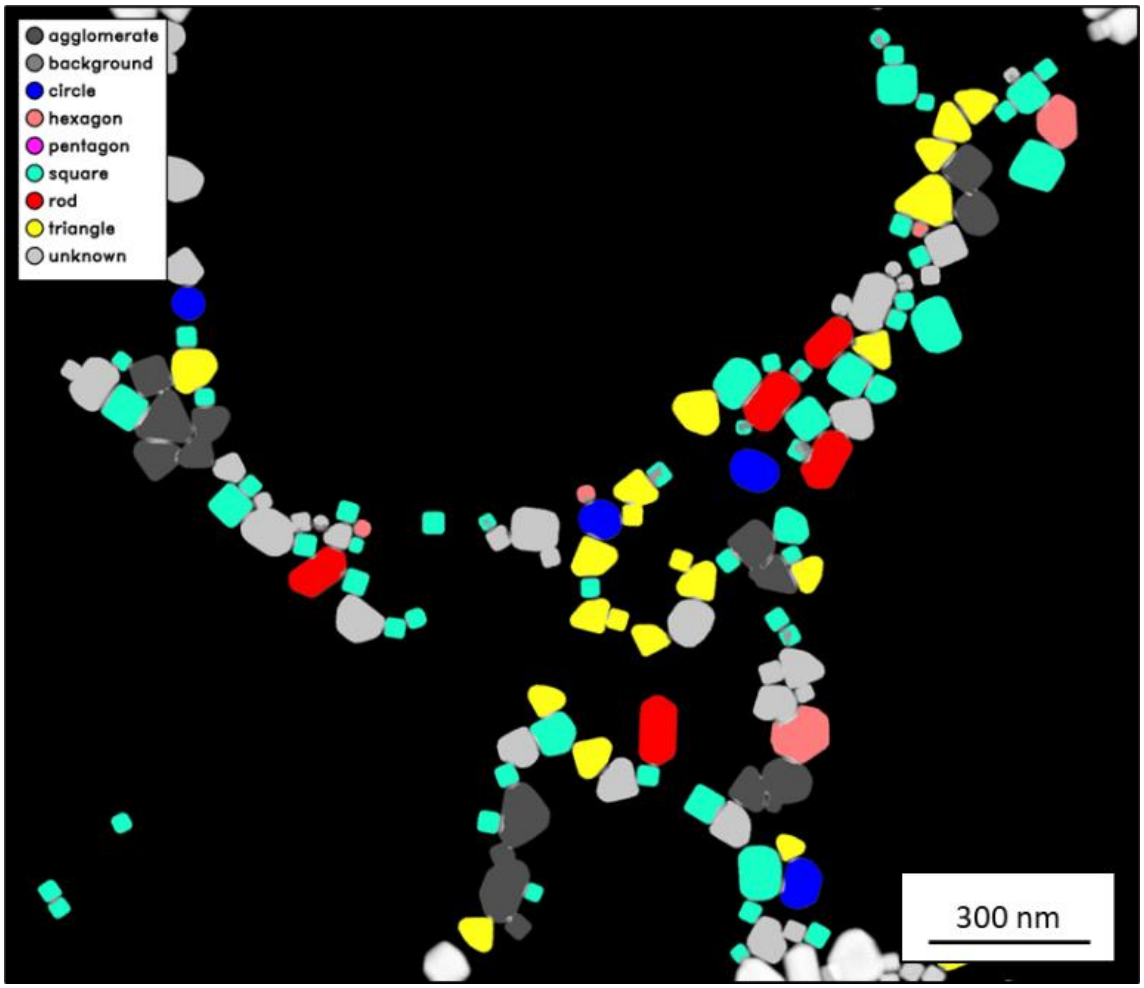


***Abbildung 52**: Auswertung von diversen Ag Nanopartikeln bei 100.000-facher Vergrößerung. Es zeigt sich, dass der Hauptteil der Nanopartikel trennbar ist. Zudem wurden die trennbaren Partikel richtig klassifiziert. In Aufnahmen wie diesen kann die Fähigkeit des Modells für diverse Nanopartikelformen verifiziert werden. In größeren Übersichtsaufnahmen mit mehreren 100 bis 1000 Partikeln ist eine verlässliche Auswertung des Menschen bezüglich verschiedener Partikelformen nicht möglich. Dies resultiert aus der unterschiedlichen Interpretation verschiedener Analysten.* [69]

Die mittlere Kantenlänge der Quadrate weicht mit 56 nm um -20 % von der menschlichen Auswertung ab. Es zeigt sich zudem, dass 2 definierte Peaks in der Größenverteilung existieren. Zum einen bei einer Kantenlänge von 60 nm, zum anderen bei einer Kantenlänge von 100 nm. In Kombination mit weiteren ausgewerteten Partikeleigenschaften wie der Zirkularität könnten so vermutlich komplexe Mechanismen der Synthese abgeleitet werden. Die mittlere Kantenlänge der Dreiecke, welche wahrscheinlich Tetraeder abbilden, beträgt 127 nm. Diese weicht um 6 % von der menschlichen Auswertung ab.

4.4.2 Au Nanopartikel bei 75.000-facher Vergrößerung (STEM)

Die Auswertung von Au Nanopartikeln ist in **Abbildung 54** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 53** & **Tabelle 8** gezeigt. Die Probe zeigt einen hellen Hintergrund / Probenträger, wodurch die Segmentierung der Partikel erschwert wird. Ziel der Synthese war es, kugelförmige Nanopartikel herzustellen. Die Auswertung der Probe erfolgt nach dem mittleren Äquivalenzdurchmesser der Partikel.

***Tabelle 8**: Auswertung der Partikelgrößen aus **Abbildung 54**sowie menschliche Auswertung aller Partikelformen.*

	Morphologie	**Ør / nm**	**n**
Mensch	Gesamt (Äquivalenzradius)	31 ± 11	50
Modell	Gesamt (Äquivalenzradius)	24 ± 9	127

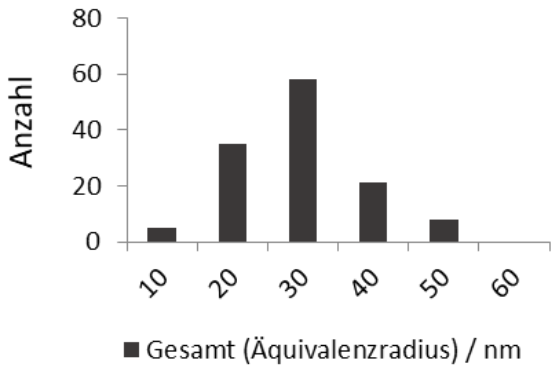


***Abbildung 53**: Auswertung der gesamten Probe aus **Abbildung 54** nach dem Äquivalenzradius.*

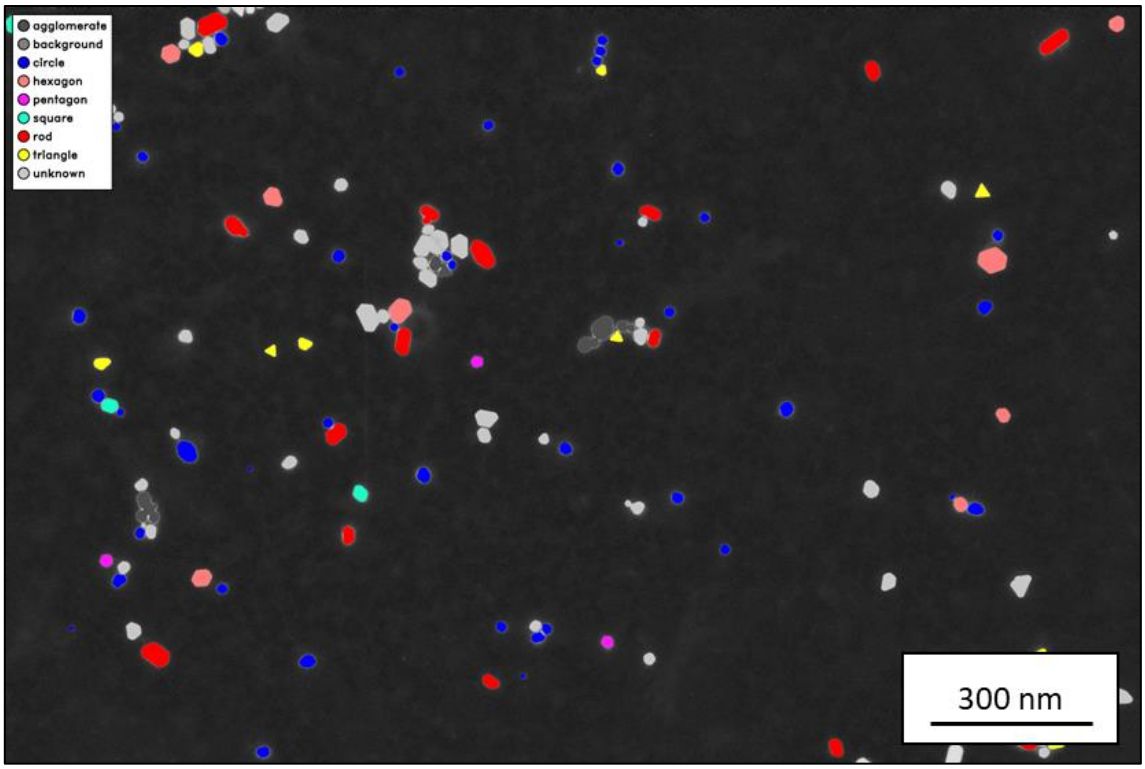


***Abbildung 54**: Auswertung von diversen Au Nanopartikeln bei 75.000-facher Vergrößerung. Trotz des geringen Intensitätsunterschiedes zwischen Partikel und Hintergrund ist die Trennung von Partikeln möglich. Die Abbildung zeigt auch, dass elongierte Partikel als Stäbchen identifiziert werden.*

Der mittlere Äquivalenzdurchmesser weicht mit 24 nm um -23 % von der menschlichen Auswertung ab. Zudem ist interessant, dass die KI elongierte Partikel als Stäbchen erkannt hat. Bei der Betrachtung des Seitenverhältnisses dieser Partikel zeigt sich, dass diese mit 1,7 ± 0,2 deutlich über dem der anderen Partikel liegen. Das durchschnittliche Seitenverhältnis der Partikel liegt bei 1,1 ± 0,1.

Die Auswertung zeigt, dass Proben, die aus menschlicher Betrachtung einer relativ monoton in der Partikelverteilung erscheinen, von der KI in zwei Spezies aufgeteilt werden können. In diesem Falle Stäbchen oder elongierte Partikel sowie Kreise, welche im 3-dimensionalen Raum Kugeln sind.

4.4.3 Au Nanopartikel bei 75.000-facher Vergrößerung II (STEM)

Die Auswertung von Au Nanopartikeln ist in **Abbildung 56** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 55** & **Tabelle 9** gezeigt. Die Probe zeigt eine wechselnde Intensität des Probenträgers. Für Fälle wie diese wurden Intensitätsbasierte Gewichtungskarten eingeführt. Wie zuvor gezeigt, können somit Partikel in den helleren Bereichen des Bildes ebenfalls verlässlich segmentiert werden. Die Auswertung der Probe erfolgt nach dem mittleren Äquivalenzdurchmesser der Partikel.

***Tabelle 9**: Auswertung der Partikelgrößen aus **Abbildung 56**sowie menschliche Auswertung aller Partikelformen.*

	Morphologie	**Ør / nm**	**n**
Mensch	Gesamt (Äquivalenzradius)	35 ± 11	60
Modell	Gesamt (Äquivalenzradius)	19 ± 5	855

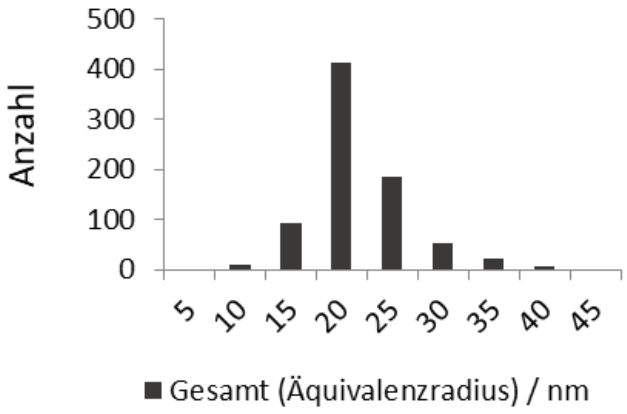


***Abbildung 55**: Auswertung der gesamten Probe aus nach dem Äquivalenzradius.*

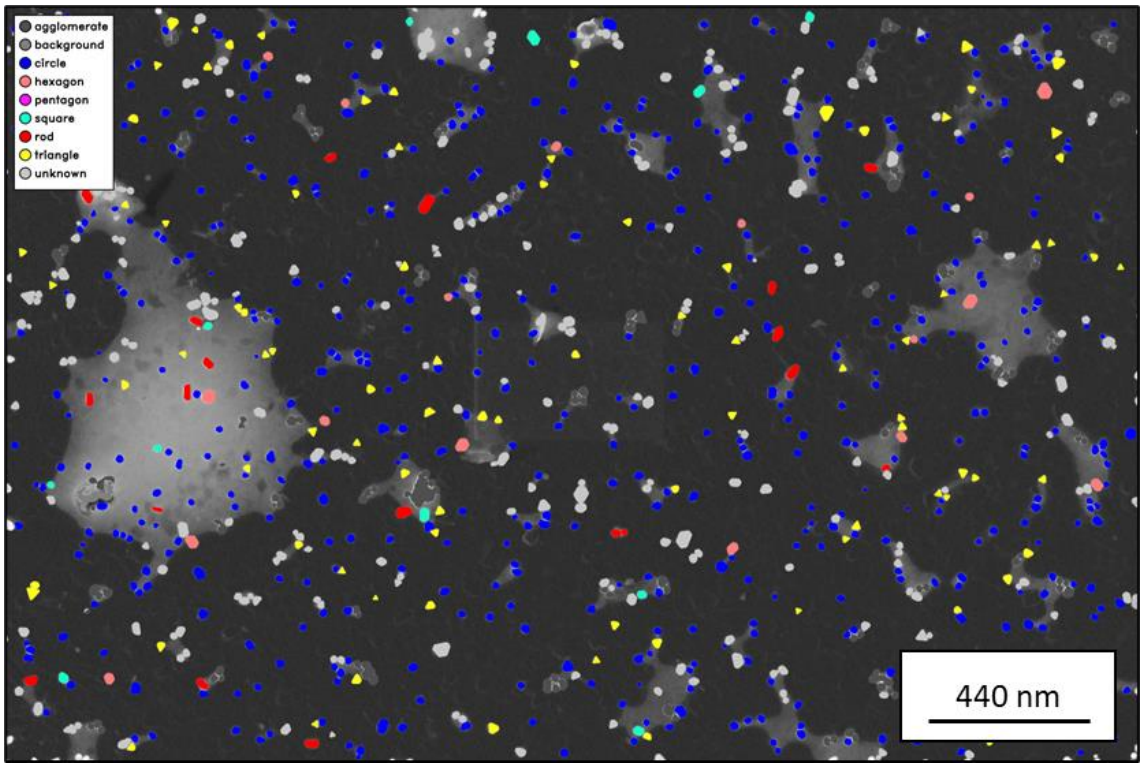


***Abbildung 56**: Auswertung von diversen Au Nanopartikeln bei 75.000-facher Vergrößerung. Das Bild zeigt, wie unterschiedliche Bereiche des Probenträgers durch Rückstände der Synthese oder des stabilisierenden organischen Zusatzes heller erscheinen.*

Die Segmentierung erfolgt verlässlich, wodurch > 800 Partikel identifiziert werden konnten. Diese wurden vom Klassifizierungsmodell anschließend überwiegend als sphärische Partikel erkannt. Der mittlere Äquivalenzdurchmesser weicht mit 19 nm um -46 % von der menschlichen Auswertung ab. Die Abweichung ist nicht signifikant.

Beeindruckend ist hier die Fähigkeit beider Modelle, die gezeigten Partikel zu verarbeiten. Der mittlere Radius eines Partikels beträgt ca. 11 ± 3 Pixel, wobei der Maßstab des Bildes 0,55 px nm^{-1} beträgt. Hierdurch zeigt sich, dass das Segmentierungsmodell in der Lage ist, Partikel über eine große Skala möglicher Partikeldurchmesser (in Pixeln) zu segmentieren. Zudem zeigen die analysierten STEM-Partikel einen geringeren Durchmesser als die vom Menschen ausgewerteten Partikel. Dieser Trend ist ebenfalls in der Literatur zu finden. [88]

4.4.4 SiO_2 Mikrokugeln bei 15.000- und 50.000-facher Vergrößerung (SE)

Die Auswertung von SiO_2 Mikrokugeln, aufgenommen im SE-Modus ist in **Abbildung 58** und **Abbildung 59** gezeigt. Beide Abbildungen zeigen dieselbe Probe in verschiedenen Vergrößerungen. Dies soll zeigen, dass die KI-Auswertung trotz unterschiedlicher Vergrößerungen des REM konstante Partikeldurchmesser erzeugt. Die menschliche Auswertung listet eine reine Kugel-Population mit einem mittleren Radius von 215±30 nm. [89] Die Partikelgrößenverteilung der Modelle ist in **Abbildung 57** gezeigt.

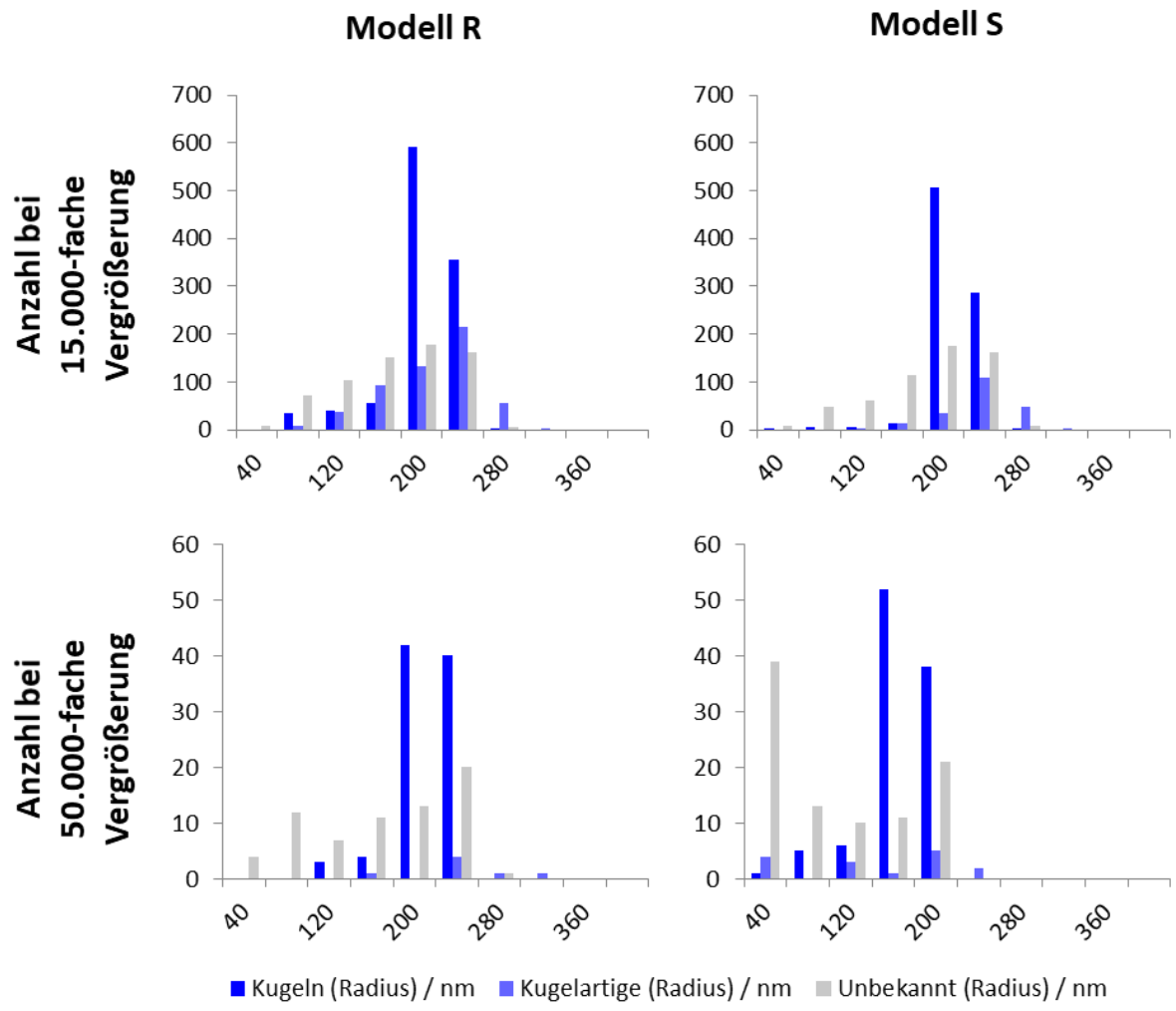


***Abbildung 57**: Partikelgrößenverteilung als Histogramme, welche in **Abbildung 58** bei 15.000-facher Vergrößerung und **Abbildung 59** in 50.000-facher Vergrößerung gezeigt sind.*

Wie in **Abbildung 57** zu erkennen ist, weist die Partikelgrößenverteilung von beiden Modellen den gleichen Trend auf. Dominant sind in allen Aufnahmen Partikel der Kategorie „Kugeln“, gefolgt von „Kugelartigen“. Ein hoher Anteil von Partikel der Klasse „Unbekannt“ weist auf die Schwierigkeit des Klassifizierungsmodells hin, „Kugeln“ von „Kugelartigen“ zu unterscheiden. Zudem sind wenige Partikel als „Würfel“ bzw. Stäbchen erkannt worden. Der Anteil der falsch klassifizierten Partikel ist jedoch gering und kann vernachlässigt werden.

Betrachtet man die Auswertung der Modelle S & R in **Abbildung 58** ist der Hauptunterschied die fehlende Segmentierung von teilweise verdeckten Partikeln. So zeigt sich, dass Partikel, die von anderen Partikeln überlagert werden, von Modell S als Hintergrund eingeteilt werden. Dies spart grundsätzlich Zeit in der folgenden Analyse, da weniger Partikel ausgeschnitten und klassifiziert werden müssen. Die Größenverteilung wird ebenfalls nicht beeinflusst. Vollständig gezeigte Partikel werden innerhalb des erwartbaren Fehlers segmentiert und klassifiziert.

***Tabelle 10**: Auswertung der Partikelgrößen aus **Abbildung 58** und **Abbildung 59**sowie der menschlichen Analyse. Ersichtlich ist die geringe Abweichung beider Modelle. Die relative Abweichung beider Modelle bezüglich des mittleren Durchmessers echter Kugeln liegt bei -11 %. Die relative Abweichung zwischen dem Durchmesser echter Kugeln bei 15.000-facher zu 50.000-facher Vergrößerungen liegt bei 0,5 %.*

	Morphologie	15.000-fache Vergrößerung		50.000-fache Vergrößerung	
		Ør / nm	n	Ør / nm	n
Mensch	*Kugeln* (Radius)	*215±30*	*100*	186*±78*	30
Modell R	Kugeln (Radius)	192 ± 18	803	193 ± 28	89
	Kugelartige (Radius)	207 ± 40	272	231 ± 47	7
	Unbekannt (Radius)	154 ± 52	682	147 ± 66	68
	Gesamt (Radius)	*188 ± 44*	*1757*	*177 ± 56*	*164*
Modell S	Kugeln (Radius)	192 ± 20	819	191 ± 26	94
	Kugelartige (Radius)	214 ± 35	211	183 ± 68	7
	Unbekannt (Radius)	163 ± 51	576	115 ± 76	86
	Gesamt (Radius)	*185 ± 40*	*1606*	*154 ± 69*	*187*

Wie **Tabelle 10** zu entnehmen, weichen die erzielten Partikeldurchmesser nur geringfügig von der menschlichen Analyse ab (-11 %). Die Abweichung ist nicht signifikant. Da der Partikeldurchmesser von der Qualität der Segmentierung anhängt, ist die Auswertung ein guter Indikator für eine gelungene Segmentierung einzelner Partikel. Bei 803 (89) Partikel der Kategorie Kugel wurden gefunden, diese Anzahl reicht, um eine statistisch relevante Aussage bezüglich der Partikelgröße zu treffen.

Abschließend kann gefolgert werde, dass Kugeln und kugelartige SiO_2 Partikel in einer 15.000- bis 50.000-facher Vergrößerung verlässlich vom KI-Ensemble ausgewertet werden können.

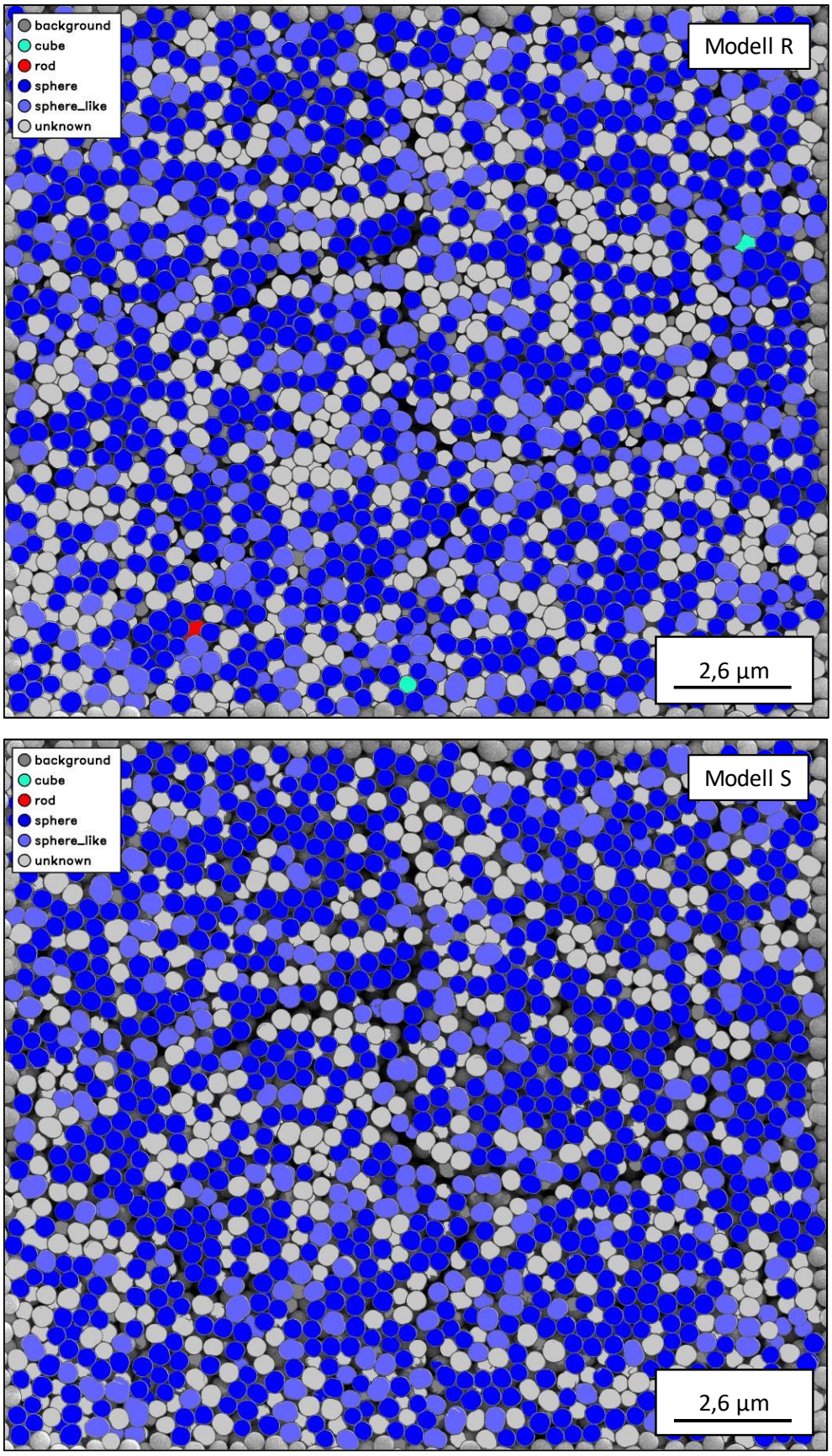


Abbildung 58: *Auswertung von SiO_2 Mikrokugeln bei 15.000-facher Vergrößerung.* [89] *Die Skalierung beträgt 8,8 nm px^{-1}.*

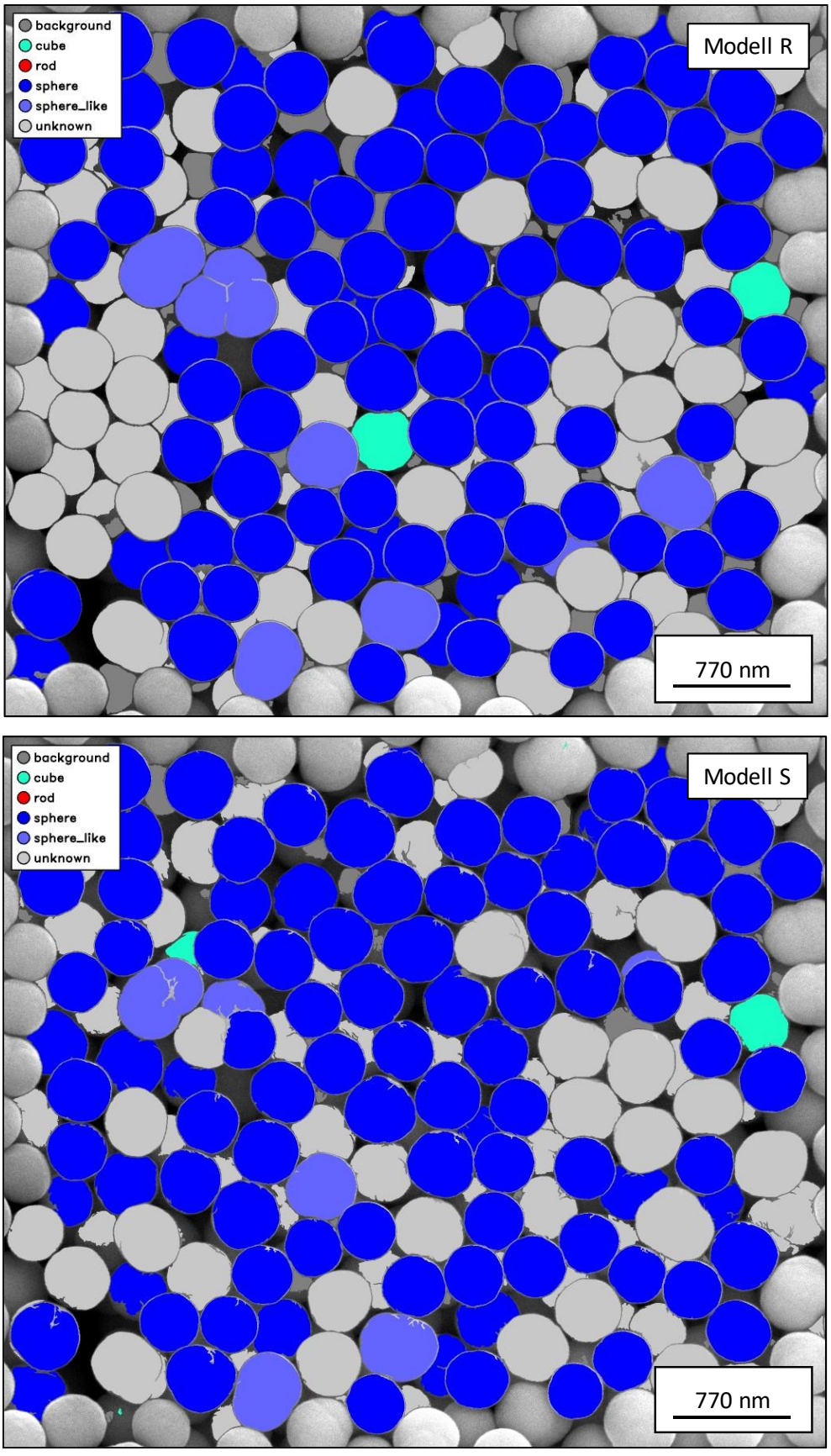


***Abbildung 59**: Auswertung von SiO_2 Mikrokugeln 50.000-facher Vergrößerung. Die Skalierung beträgt 2,64 nm px^{-1}.*

4.4.5 ZnO Mikrokugeln bei 10.000-facher Vergrößerung (SE)

Die Auswertung von ZnO Mikrokugeln ist in **Abbildung 61** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 60** & **Tabelle 11** gegeben. Es zeigt die Fähigkeit beider Modelle, sich an variable kugelförmige Objekte anzupassen. Das Bild zeigt ein höheres Rauschen als **Abbildung 58**, wie am Hintergrund zu erkennen ist. Wie bereits in der Auswertung des gesamten Datensatzes gezeigt, besteht ein geringer Zusammenhang zwischen dem Signal-zu-Rauschen-Verhältnis und der Segmentierungsleistung des Modell R.

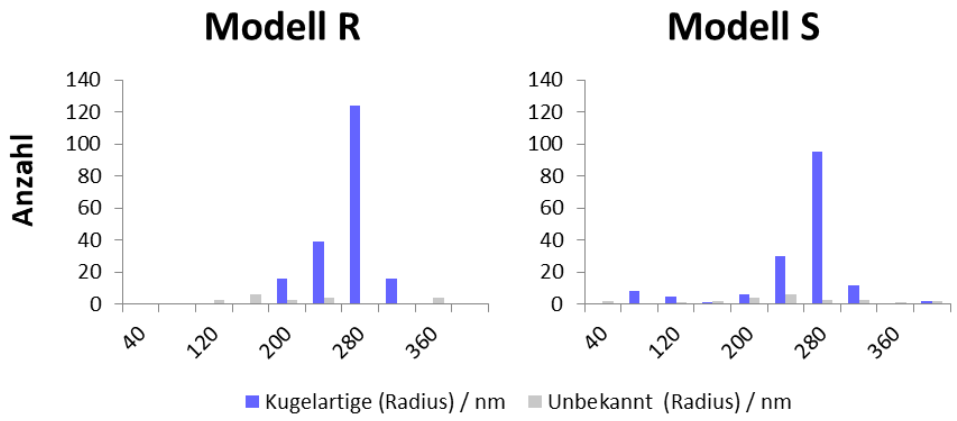


***Abbildung 60**: Partikelgrößenverteilung als Histogramme, welche in **Abbildung 61** bei 10.000-facher Vergrößerung gezeigt ist.*

***Tabelle 11**: Auswertung der Partikelgrößen von ZnO Mikropartikeln aus **Abbildung 61**sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	Ør / nm	n
Mensch	Kugelartige (Radius)	250 ± 51	100
Modell R	Kugelartige (Radius)	247 ± 27	195
	Unbekannt (Radius)	193 ± 84	20
	Gesamt (Radius)	*243 ± 39*	*215*
Modell S	Kugelartige (Radius)	240 ± 52	152
	Unbekannt (Radius)	231 ± 82	16
	Gesamt (Radius)	*236 ± 55*	*168*

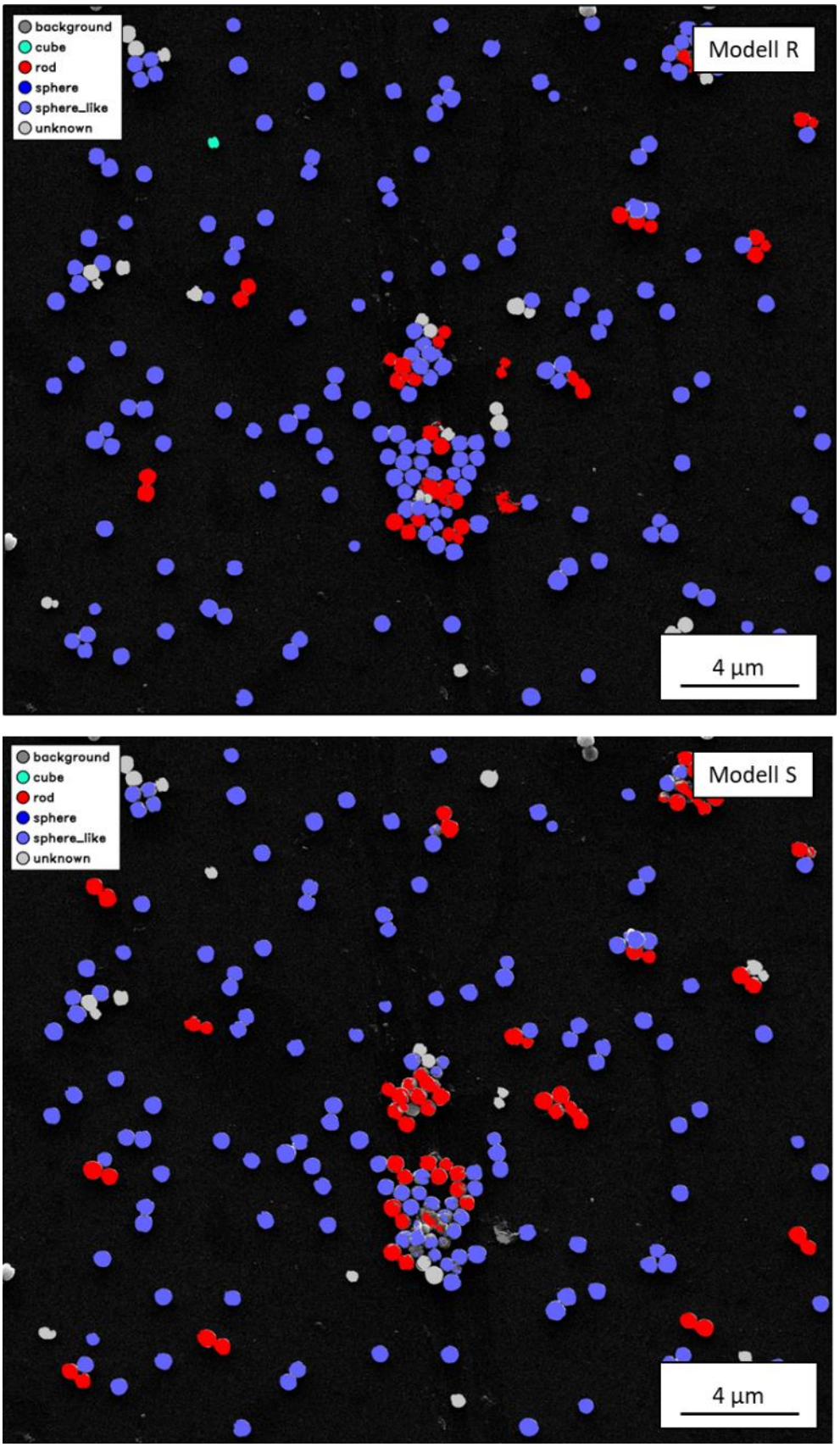


***Abbildung 61**: Automatische Auswertung von ZnO Mikropartikeln bei 10.000-facher Vergrößerung. REM-Aufnahme aus [89] Die Skalierung beträgt 13,2 nm px^{-1}.*

Im Vergleich schneiden die Segmentierungs-Modelle ähnlich ab, wobei Modell R einen besseren Grad der Segmentierung erreicht als Modell S. Somit kann Modell R mehr Partikel trennen.

Die Modelle zeigen beide denselben Trend mit dominanter Menge an identifizierten kugelartigen Objekten. Modell S zeigt eine geringere Fähigkeit zur Trennung der Partikel, wodurch insgesamt 43 Kugelartige Objekte weniger identifiziert wurden als bei Modell R. Fälschlicherweise als Stäbchen identifizierte Partikel sind nicht gezeigt. Der menschliche Analyst kann diese offensichtliche falsche Auswertung ignorieren. Dies zeigt allerdings die Schwäche des Systems in der Hinsicht auf. Es macht deutlich, dass das System lediglich als Unterstützung für den menschlichen Analysten dienen darf, da komplexe Sicherungen nicht existieren. Das ist ein grundsätzlicher Punkt, der in weiteren Versionen des Workflows adressiert werden sollte.

Die gemessene Partikelgröße variiert geringfügig. Modell R zeigt eine Abweichung von -1 % zur menschlichen Auswertung, während Modell S eine Abweichung von -4 % zeigt. Keine Abweichung ist signifikant.

4.4.6 TiO2 Submikrokugeln bei 50.000-facher Vergrößerung (SE)

Die Auswertung von TiO2 (Anatas) Submikrokugeln ist in **Abbildung 63** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 62** & **Tabelle 13** gegeben. Die Kugeln sind deutlich kleiner als die bisher gezeigten SiO2 und ZnO Partikel.

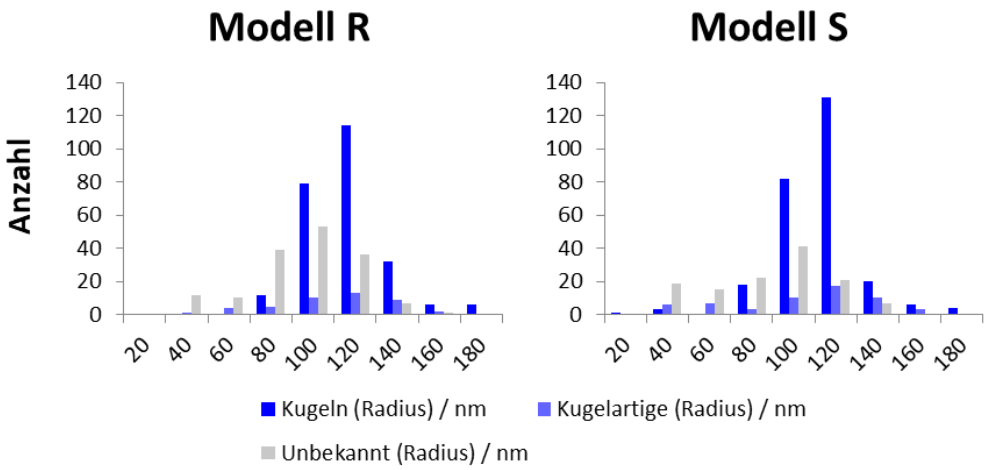


***Abbildung 62**: Partikelgrößenverteilung als Histogramme, welche in **Abbildung 63** bei 50.000-facher Vergrößerung gezeigt sind.*

***Tabelle 12**: Auswertung der Partikelgrößen von TiO_2 Mikropartikeln aus **Abbildung 63**sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	Ør / nm	n
Mensch	Kugeln (Radius)	110 ± 20	100
Modell R	Kugeln (Radius)	107 ± 18	249
	Kugelartige (Radius)	101 ± 28	44
	Unbekannt (Radius)	85 ± 24	158
	Gesamt (Radius)	*98 ± 24*	*451*
Modell S	Kugeln (Radius)	104 ± 218	257
	Kugelartige (Radius)	96 ± 233	48
	Unbekannt (Radius)	78 ± 229	117
	Gesamt (Radius)	*93 ± 26*	*422*

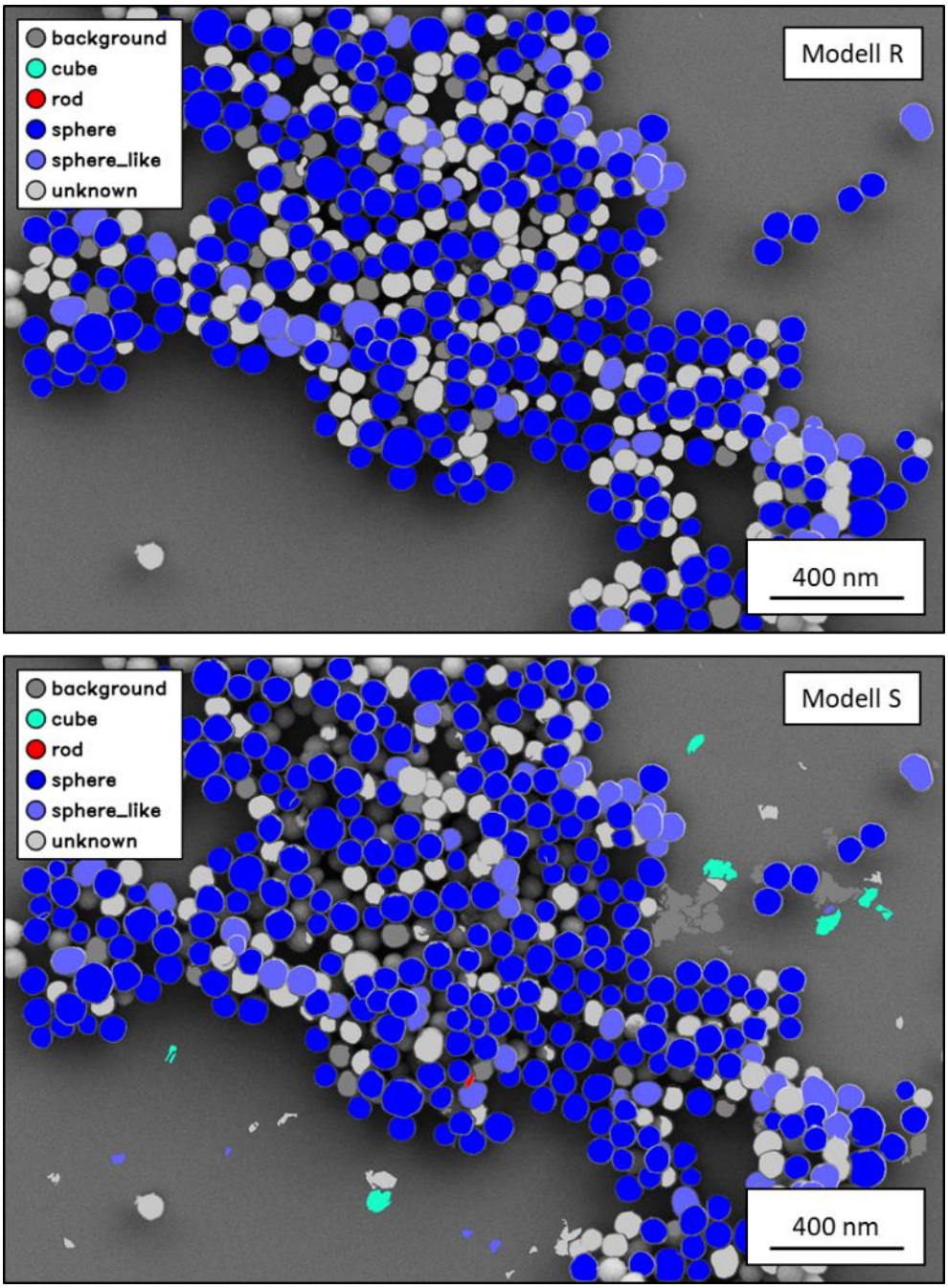


***Abbildung 63**: Automatische Auswertung von TiO_2 Submikropartikeln. REM-Aufnahme aus [89] Die Skalierung beträgt 0,66 nm px^{-1}.*

Im Vergleich schneiden die Segmentierungs-Modelle ähnlich ab, wobei Modell R einen besseren Grad der Segmentierung erreicht als Modell S. Somit kann Modell R mehr Partikel trennen. Zudem fällt ein hoher Anteil falsch segmentierten Hintergrundes auf.

Hintergrund, der von den Segmentierungsmodellen fälschlicherweise als Vordergrund erkannt wurde, ist anschließend von dem Klassifizierungsmodell in eine der möglichen Partikelformen identifiziert worden. Das Klassifizierungsmodell konnte in den als Würfel

(türkis) erkannten Bereichen keine typischen Hintergrund-Strukturen identifizieren. Da Hintergrund lediglich das in REM-Bildern omnipräsente Gauß‘sche Rauschen abbildet, hat die KI „geraten“. Dies ist ein offensichtliches Fehlverhalten des Klassifizierungsmodells. Eine Vergrößerung des Datensatzes oder das Einfügen stärkeren Rauschens könnte das Modell verbessern.

Die Größenverteilung zeigt eine geringe Abweichung zwischen menschlicher Auswertung und KI-Systemen. Bezüglich der mittleren Größe von echten Kugeln weicht Modell R um -3 % und Modell S um -6 % von der menschlichen Auswertung ab. Keine Abweichung ist signifikant. Interessant ist in diesem Bild eindeutige Zuordnung von Objekten in die Klassen Kugeln und Kugelartige. Die Zirkularität von Kugeln liegt bei 0,98 ± 0,02, wobei diese für kugelartige Objekte bei 0,91 ± 0,07. Diese Unterscheidung ist nicht signifikant.

4.4.7 SiO_2 Nanopartikel bei 150.000-facher Vergrößerung (SE)

Die Auswertung von SiO_2 Nanopartikeln ist in **Abbildung 65** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 64** & **Tabelle 13** gezeigt. Die Kugeln sind erneut kleiner als die bisher gezeigten Partikel.

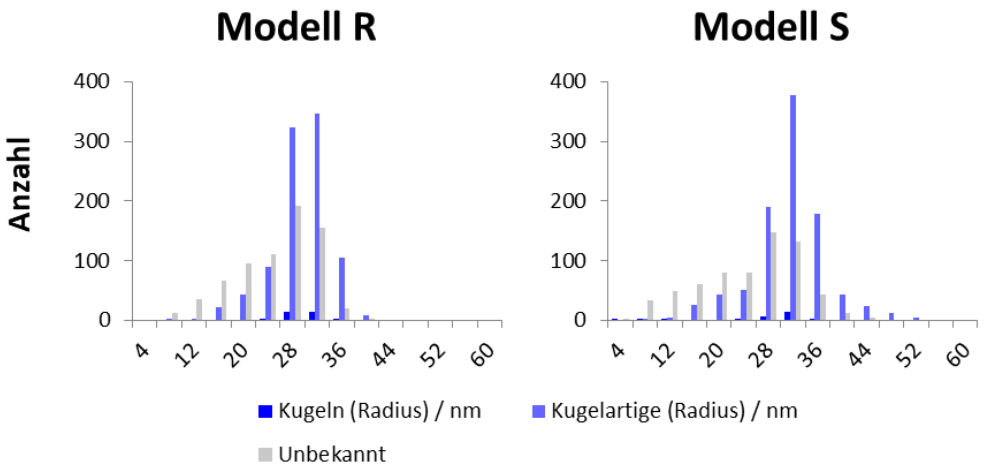


***Abbildung 64**: Partikelgrößenverteilung als Histogramme der Partikel, welche in **Abbildung 65** bei 150.000-facher Vergrößerung gezeigt sind.*

***Tabelle 13**: Auswertung der Partikelgrößen von TiO_2 Mikropartikeln aus **Abbildung 65**sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	Ø / nm	n
Mensch	Kugelartige (Radius)	30 ± 4	100
Modell R	Kugeln (Radius)	28 ± 3	34
	Kugelartige (Radius)	27 ± 5	940
	Unbekannt (Radius)	23 ± 7	689
	Gesamt (Radius)	*26 ± 6*	*1663*
Modell S	Kugeln (Radius)	27 ± 7	20
	Kugelartige (Radius)	30 ± 6	945
	Unbekannt (Radius)	23 ± 8	641
	Gesamt (Radius)	*27 ± 8*	*1606*

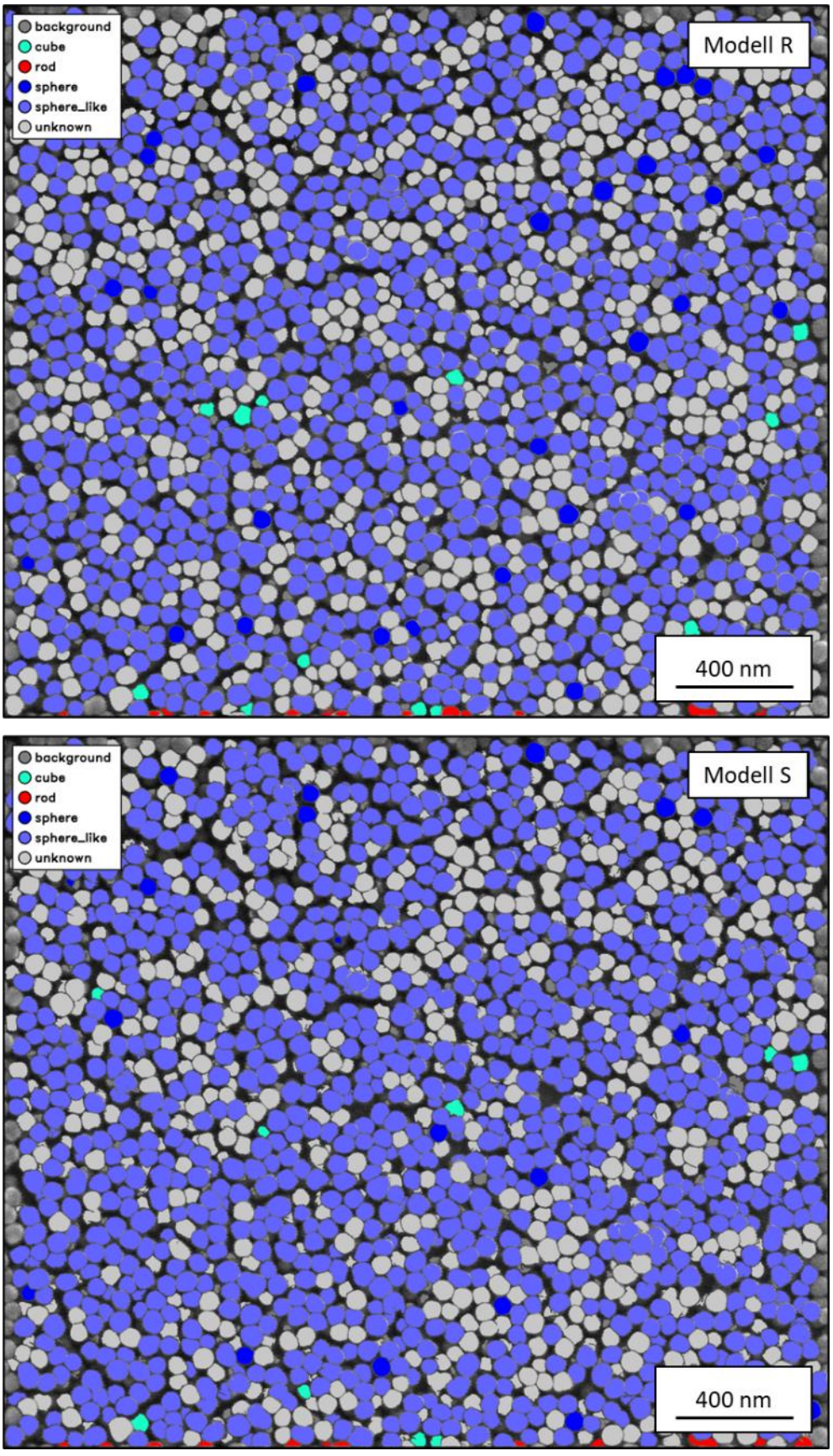


***Abbildung 65**: Automatische Auswertung von SiO_2 Nanopartikeln. REM-Aufnahme aus [89] Die Skalierung beträgt 1,32 nm px^{-1}.*

Im Vergleich schneiden die Segmentierungs-Modelle ähnlich ab, wobei Modell R einen höheren Grad der Segmentierung erreicht als Modell S. Somit kann Modell R ca. 60 Partikel mehr erkennen. Bei einer Anzahl von über 1600 erkannten Partikeln ist dies jedoch nicht weiter von belangen.

Die Analyse von sphärischen Nanopartikeln mittels KI-Systemen zeigt einen hohen Grad der Segmentierung für Modell R wie auch für Modell S. Zudem werden die Partikel sicher als kugelartige Objekte erkannt. Dass die Partikel als Kugelartige und nicht als echte Kugeln erkannt werden, liegt wahrscheinlich und dem hohen Rauschen des Bildes und dem unscharfen Auftreten der Partikel. Die Unschärfe ist bedingt durch die hohe Vergrößerung. Es zeigt auch die Fähigkeit aller Modelle, mit unscharfen Bildern zu arbeiten und dennoch eine verlässliche Analyse der Partikelform und Größe zu gewährleisten. Des Weiteren ist interessant, dass am unteren Bildrand eine hohe Anzahl von Objekten als Stäbchen identifiziert wurde. Wodurch es in diesem Bereich zu dieser Verwechslung kam, ist unklar. Durch die selektive Auswertung des Workflows wird der menschliche Analyst dennoch sicher in der Lage sein, die Partikeldurchmesser der kugelartigen Partikel zu bestimmen. Selbiges gilt für die geringe Anzahl fälschlicherweise bestimmter Würfel.

Die Partikel besitzen einen Radius von 28 ± 3 (Modell R) und 27 ± 7 (Modell S) was einer Abweichung von -7 % (Modell R) und -10 % (Modell S) entspricht. Die mittlere Zirkularität aller als kugelartige erkannten Partikel beträgt 0,94 ± 0,05. Die segmentierten Partikel zeigen teilweise einen welligen Rand, der wahrscheinlich durch das Rauschen verursacht wurde. Es kann gefolgert werden, dass die Auswertung stabil gegen diese Form von Rauschen ist.

4.4.8 Ag Nanopartikel undefinierter Form bei unbekannter Vergrößerung (SE)

Die Auswertung von Ag Nanopartikeln bei unbekannter Vergrößerung ist in **Abbildung 66** gezeigt. Die zugehörige Partikelgrößenverteilung ist in **Abbildung 66** & **Tabelle 14** gezeigt. Die Probe zeigt einen dominanten Anteil kugelartiger und deformierter Objekte.

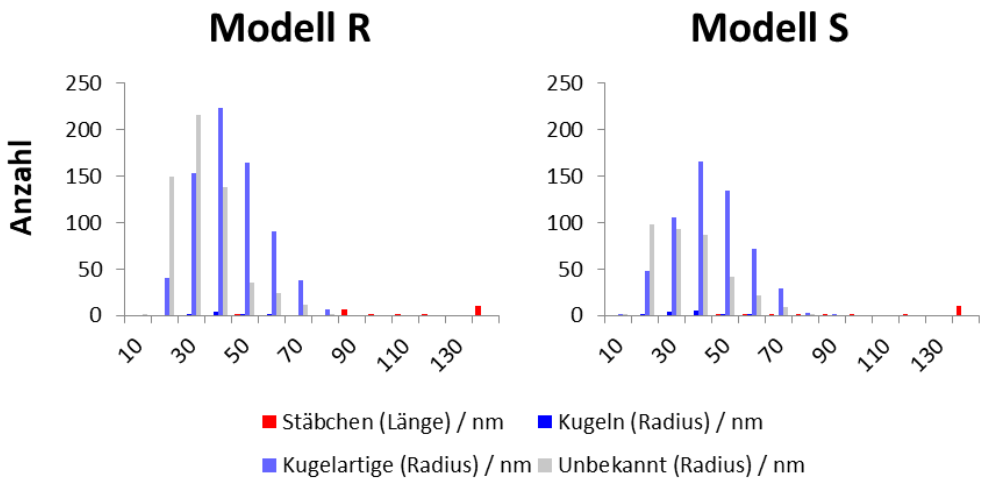


***Abbildung 66**: Partikelgrößenverteilung als Histogramme, welche in **Abbildung 67** gezeigt sind.*

***Tabelle 14**: Auswertung der Partikelgrößen von Ag Nanopartikeln aus **Abbildung 67** sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	Ø / nm	n
Mensch	Kugelartige (Radius)	55 ± 23	ca. 100
Modell R	Stäbchen (Länge)	125 ± 58	22
	Stäbchen (Breite)	59 ± 20	
	Kugeln (Radius)	40 ± 12	8
	Kugelartige (Radius)	38 ± 13	717
	Unbekannt (Radius)	28 ± 12	578
	Gesamt (Radius)	36 ± 30	747
Modell S	Stäbchen (Länge)	129 ± 52	18
	Stäbchen (Breite)	59 ± 20	
	Kugeln (Radius)	34 ± 12	13
	Kugelartige (Radius)	38 ± 13	561
	Unbekannt (Radius)	30 ± 13	354
	Gesamt (Radius)	37 ± 20	946

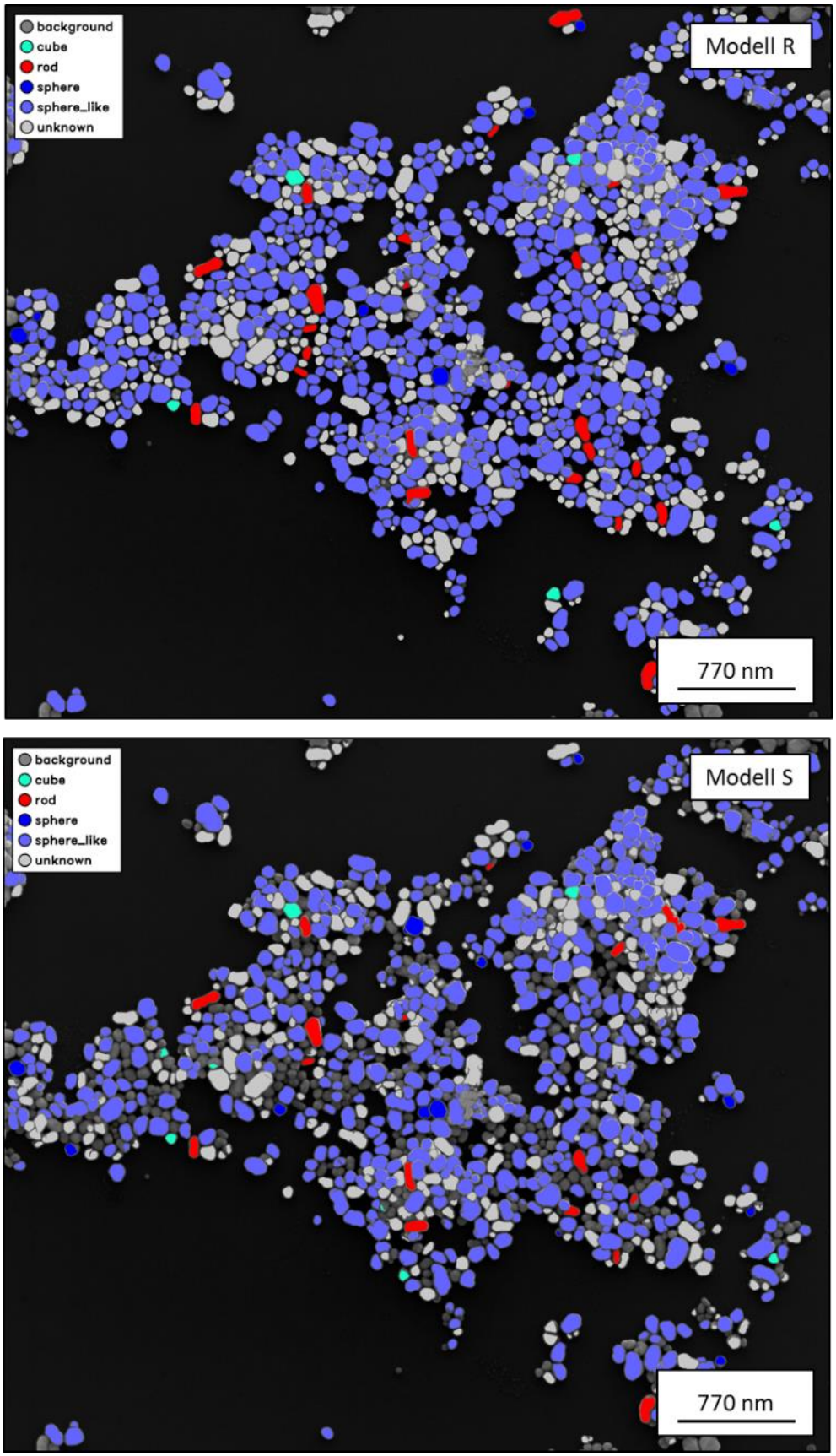


***Abbildung 67**: Automatische Auswertung von Ag Nanopartikeln. Die Skalierung beträgt 5,41 nm px-1.*

Beide Modelle segmentieren das Bild in ausreichendem Maße, das Partikel identifizierbar werden. Modell S vernachlässigt hierbei eine Vielzahl von Objekten von kleineren Partikeln, die zwischen größeren Liegen. Die Probe zeigt einen dominanten Anteil von kugelartigen Partikeln. Zusätzlich werden eine Vielzahl von Stäbchen identifiziert. Einige Würfel werden ebenfalls identifiziert. Bei Betrachtung der „Würfel“ lässt sich feststellen, dass diese Einschätzung des Modells offenkundig falsch ist. Die Klassifizierung der Stäbchen hingegen lässt sich durch Betrachtung des Länge/Breite Verhältnisses bestätigen. Für als Stäbchen identifizierte Partikel liegt dieses bei 2,16 ± 0,47, während der durchschnittliche Wert der Probe bei 1,09 ± 0,09 liegt. Dies zeigt eine signifikante Abweichung, wodurch sich die als Stäbchen erkannten Partikel eindeutig vom Rest der Probe abgrenzen. Interessant ist, dass der menschliche Analyst der Probe diesen Sachverhalt nicht erwähnte.

Die Durchmesser hingegen zeigen den bisher erkannten Trend, dass die von der KI segmentierten Partikel einen kleineren Durchmesser aufweisen als die vom Menschen ausgewerteten. Der mittlere Partikeldurchmesser für und Kugelartige liegt bei 38 ± 13 nm (Modell R & Modell S) und weicht somit -31 % von der menschlichen Analyse ab.

4.4.9 Ag Würfel / Stäbchen bei 50.000-facher Vergrößerung (SE)

Die Auswertung einer gemischten Population von Ag-Nanopartikeln ist in **Abbildung 69** gezeigt. Die Größenauswertung aller in der Abbildung vorhandenen Partikel ist in **Tabelle 15** gezeigt. Die Histogramme der Größenverteilung sind in **Abbildung 68** dargestellt.

***Tabelle 15**: Auswertung der Partikelgrößen von Ag Nanopartikeln aus **Abbildung 67**sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	Ø / nm	n
Mensch	Würfel (Kantenlänge)	98 ± 14	100
Modell R	Würfel (Kantenlänge)	94 ± 17	50
	Stäbchen (Länge)	322 ± 266	84
	Stäbchen (Breite)	61 ± 36	
	Kugeln (Radius)	38 ± 8	9
	Kugelartige (Radius)	38 ± 16	45
	Unbekannt (Radius)	34 ± 14	166
Modell S	Würfel (Kantenlänge)	94 ± 14	46
	Stäbchen (Länge)	281 ± 255	86
	Stäbchen (Breite)	56 ± 24	
	Kugeln (Radius)	37 ± 10	12
	Kugelartige (Radius)	44 ± 17	36
	Unbekannt (Radius)	35 ± 15	119

Nach der Auswertung der REM-Aufnahme mittels der Modelle R und S zeigt sich eine ähnliche Anzahl verschiedener Partikelformen. Die mittlere Größe der einzelnen erkannten Formen unterscheiden sich nicht signifikant voneinander. Das Hauptaugenmerk der Analyse lag auf der Erstellung von Würfeln definierter Kantenlänge. Der menschliche Analyst wertete dementsprechend nur Würfel aus. Die mittlere Kantenlänge der menschlichen Auswertung beider Modelle beträgt 94 nm, was einer Abweichung von -4 % entspricht.

Weiterhin werden Stäbchen von dem KI-System in weiten Teilen vollständig segmentiert. In manchen Bereichen kommt es zu einer fehlerhaften Segmentierung. Die Stäbchen sind dann

nicht vollständig trennbar und verzerren die Größenverteilung. Dies macht sich bei der Verteilung der Partikelbreite im Bereich von 120-140 nm bemerkbar.

Ein Großteil der Partikel ist trennbar, dies ist vor allem sichtbar im zentralen Bildteil, in dem die Partikel gehäuft vorliegen. Die Einteilung der gerundeten Würfel ist jedoch nicht exakt. Abgestumpfte oder gerundete Würfel wurden vom Klassifizierungsmodell des Workflows mehrheitlich als „Kugelartige“ klassifiziert. Das Modell folgt damit dem menschlichen Verhalten, das bereits in der Umfrage zur Partikelformen beobachtet werden konnte [69] Auch der menschliche Analyst dieses Bildes analysierte lediglich „ausgereifte“ Partikel; das heißt Würfel mit ausgeprägter Eckenbildung.

Zudem muss angemerkt werden, dass die Abbildung eine Population von Tetraedern enthält. Diese werden aufgrund des Mangels einer eigenen Tetraeder-Klasse des Klassifizierungsmodells als Würfel identifiziert. Dies schmälert die Genauigkeit der errechneten mittleren Kantenlänge aller Würfel, da diese auf der Fläche der als Würfel erkannten Partikel abgeleitet wird. Die Einführung einer separaten Tetraeder-Klasse für das Klassifizierungsmodell war aufgrund der geringen Anzahl von Tetraedern im Gesamtdatensatz des REMs nicht möglich. Der Gesamtdatensatz ist in diesem Falle die Summe von einzelnen Partikeln, die aus REM-Bildern ausgeschnitten wurden. Die Größe betrug mehrere 100.000 Partikel.

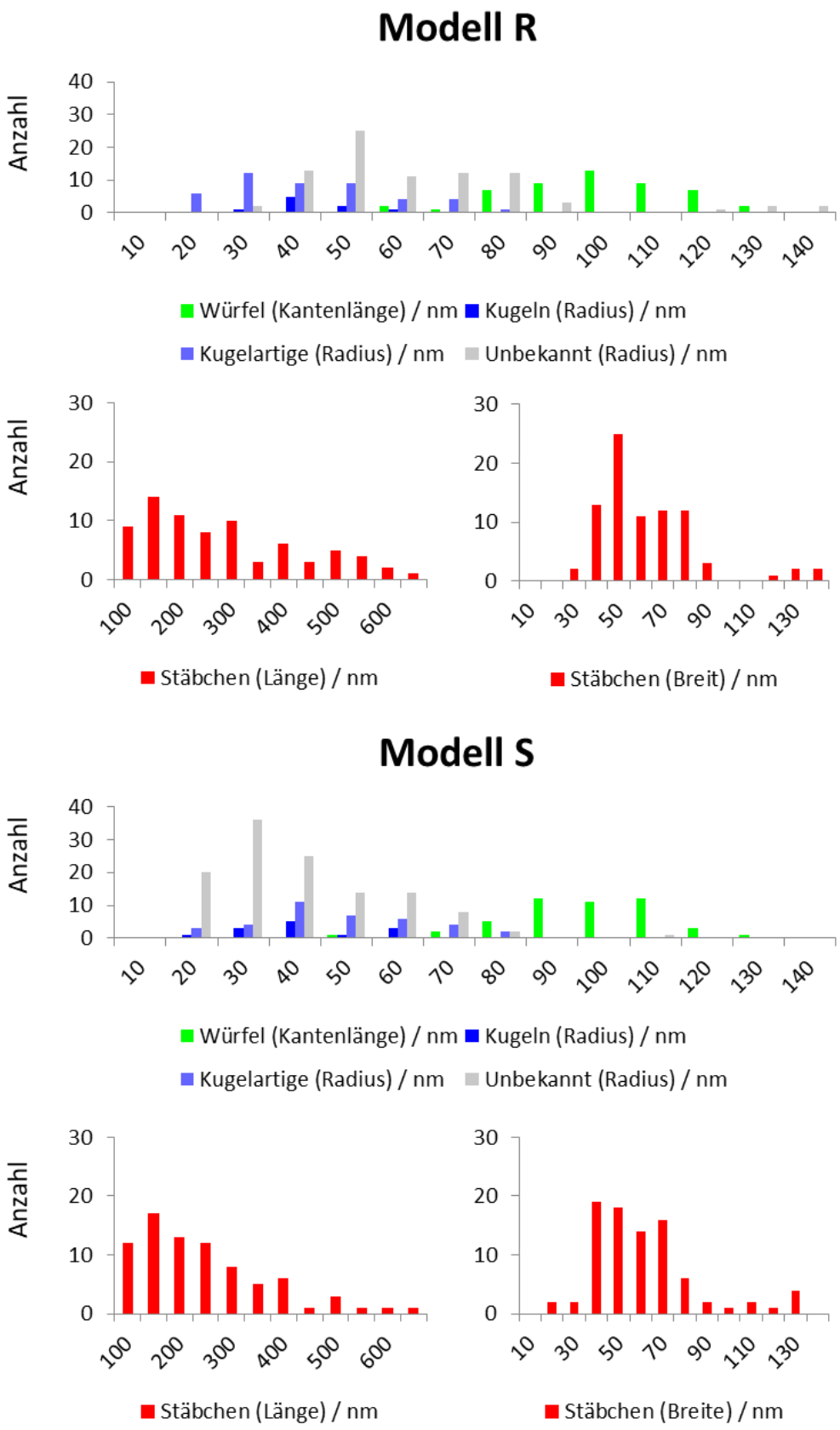


***Abbildung 68**: Partikelgrößenverteilung als Histogramme der Partikel, welche in Abbildung 69 bei 50.000-facher Vergrößerung gezeigt sind.*

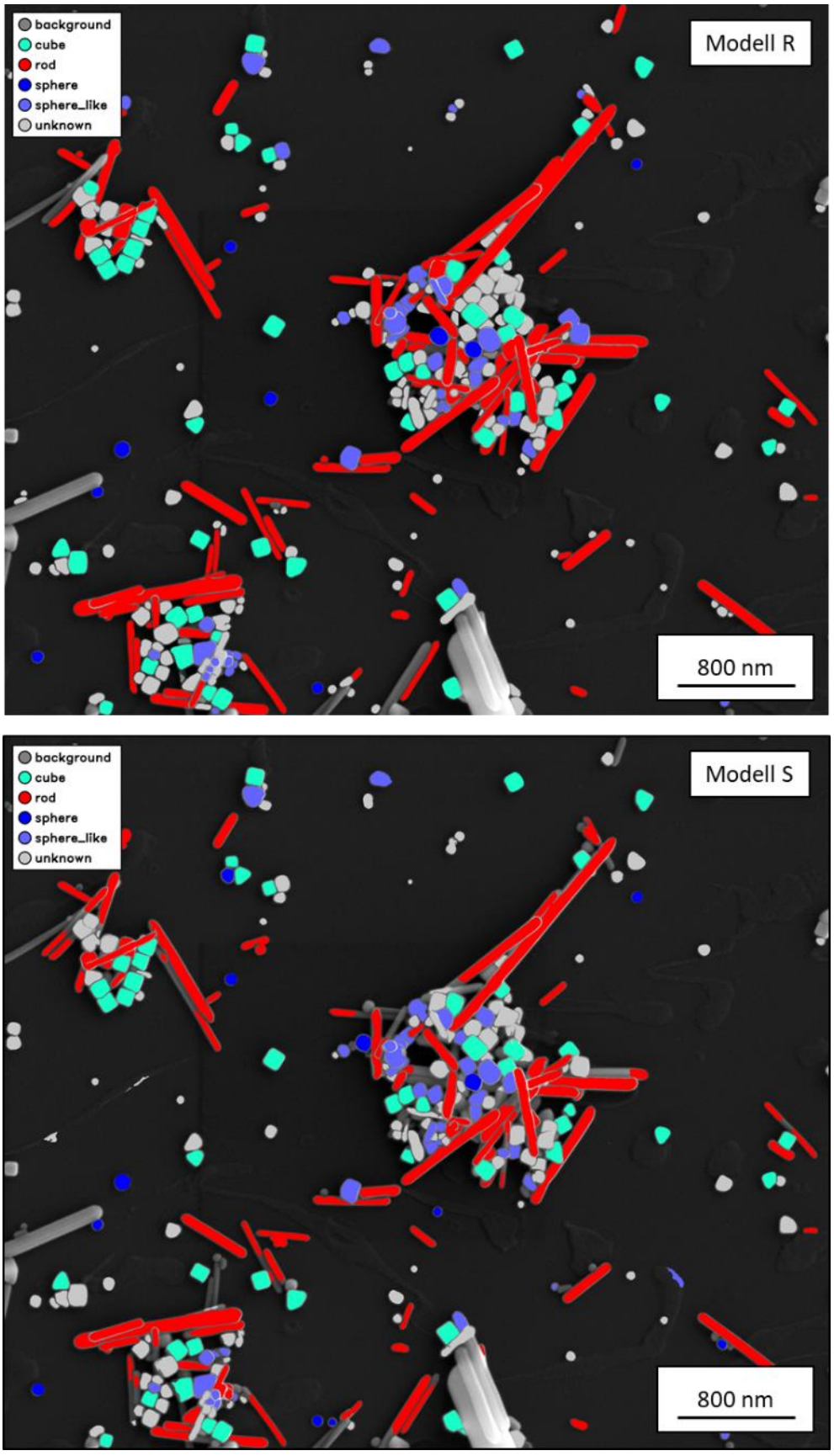


***Abbildung 69**: Automatische Auswertung von Ag Nanopartikeln. Die Skalierung beträgt 2,7 nm px-1.*

4.4.10 Ag Würfel / Stäbchen bei 100.000-facher Vergrößerung (SE)

Die Auswertung einer gemischten Population von Ag-Nanopartikeln ist in **Abbildung 71** gezeigt. Die Größenauswertung aller in der Abbildung vorhandenen Partikel ist in **Tabelle 16** gezeigt. Die Histogramme der Größenverteilung sind in **Abbildung 70** dargestellt.

***Tabelle 16**: Auswertung der Partikelgrößen von Ag Nanopartikeln aus **Abbildung 71**sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	**Ø / nm**	**n**
Mensch	Würfel (Kantenlänge)	111 ± 14	ca. 20
Modell R	Würfel (Kantenlänge)	114 ± 15	37
	Stäbchen (Länge)	382 ± 259	67
	Stäbchen (Breite)	80 ± 57	
	Kugeln (Radius)	52 ± 12	10
	Kugelartige (Radius)	60 ± 5	4
	Unbekannt (Radius)	48 ± 21	74
Modell S	Würfel (Kantenlänge)	108 ± 22	39
	Stäbchen (Länge)	319 ± 278	78
	Stäbchen (Breite)	64 ± 49	
	Kugeln (Radius)	48 ± 16	10
	Kugelartige (Radius)	72 ± 19	6
	Unbekannt (Radius)	37 ± 22	84

Dominant in der Abbildung ist die Anzahl der abgeschnittenen Stäbchen, wodurch die Gesamtanzahl der identifizierten Partikel relativ gering ist. Die Identifizierung von vollständigen Partikeln ist jedoch notwendig, um die Partikelform sicher bestimmen zu können. Die Würfel in dieser Probe sind deutlich kleiner als die abgebildeten Stäbchen. Somit scheint es, als ob die Würfel in die Zwischenräume von nebeneinanderliegenden Stäbchen „rutschen“. Dies führt zu einer geringeren mittleren Intensität der einzelnen Partikel im Bezug zum gesamten Bild. Zudem zeigen viele Partikel erneut gerundete Ecken. Das Klassifizierungs-Modell wird dadurch leicht beeinflusst, sodass viele Partikel.

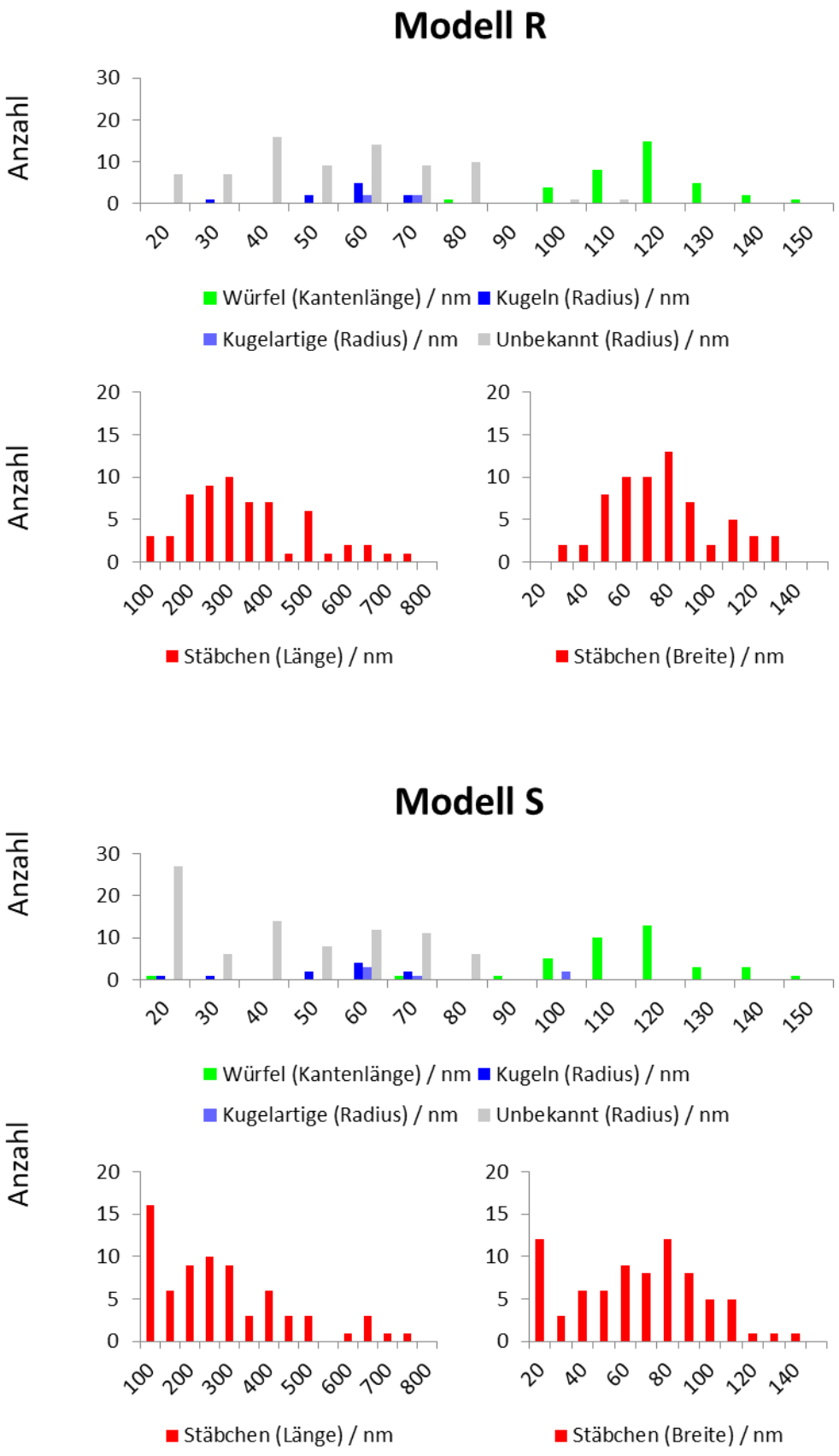


***Abbildung 70**: Partikelgrößenverteilung als Histogramme der Partikel, welche in **Abbildung 71** bei 50.000-facher Vergrößerung gezeigt sind.*

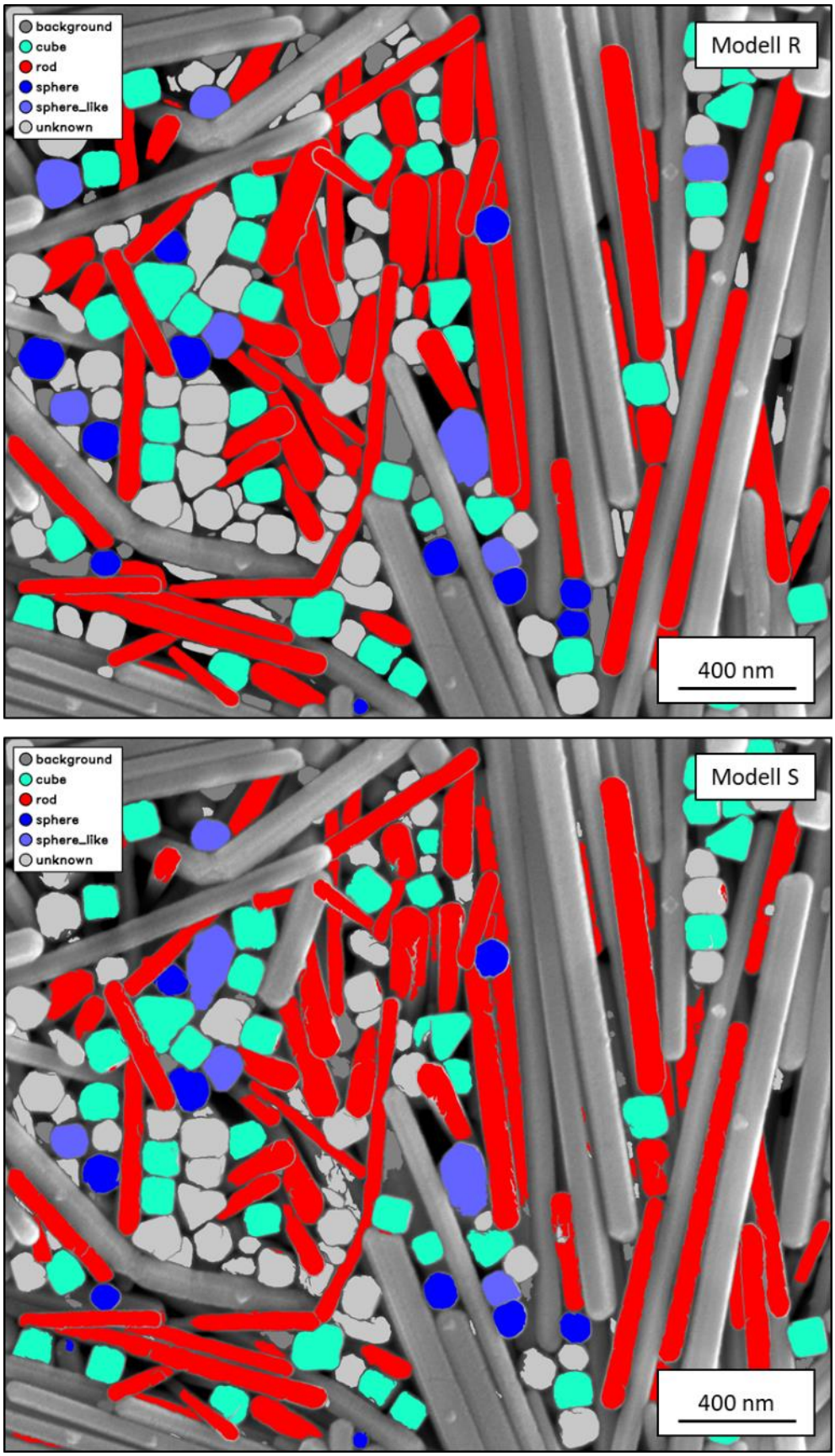


***Abbildung 71**: Automatische Auswertung von Ag Nanopartikeln. Die Skalierung beträgt 1,35 nm px-1.*

Die mittlere Größe der erkannten Würfel beträgt für das Modell R 114 nm und weicht somit 3 % von der menschlichen Auswertung ab. Modell S hingegen weicht mit einer mittleren Kantenlänge von -3 % von der menschlichen Analyse ab.

Hervorzuheben ist an dieser Stelle die Segmentierung des Modell S wie in **Abbildung 71** zu erkennen. Diese zeigt eine marmorierte Struktur an den Rändern von Partikeln, wodurch die Fläche der Partikel abnimmt. Da die Kantenlänger der Partikel in Abhängigkeit zur klassifizierten Morphologie und der Fläche der Partikel errechnet wird, kann dies für den Unterschied der Kantenlängen der Modelle verantwortlich sein. Warum diese Struktur in dem hier vorliegenden Bild auftrat, kann nicht abschließend geklärt werden. Beide Abweichungen der mittleren Kantenlänge von der menschlichen Auswertung sind nicht signifikant.

4.4.11 Ag Würfel bei 100.000-facher Vergrößerung (SE)

Die Auswertung einer annähernd reinen Probe von Ag-Nanowürfeln ist in **Abbildung 73** gezeigt. Die Größenauswertung aller in der Abbildung vorhandenen Partikel ist in **Tabelle 17** gezeigt. Die Histogramme der Größenverteilung sind in **Abbildung 72** dargestellt.

***Tabelle 17**: Auswertung der Partikelgrößen von Ag Nanopartikeln aus **Abbildung 73**sowie menschliche Auswertung der dominanten Partikelformen.*

	Morphologie	Ø / nm	n
Mensch	Würfel (Kantenlänge)	108 ± 8	100
Modell R	Würfel (Kantenlänge)	96 ± 16	321
	Stäbchen (Länge)	347 ± 355	6
	Stäbchen (Breite)	77 ± 60	
	Kugelartige (Radius)	76 ± 33	14
	Unbekannt (Radius)	41 ± 20	98
Modell S	Würfel (Kantenlänge)	92 ± 20	332
	Stäbchen (Länge)	169 ± 237	15
	Stäbchen (Breite)	48 ± 58	
	Kugelartige (Radius)	73 ± 31	14
	Unbekannt (Radius)	35 ± 21	104

Die Auswertung zeigt einen hohen Grad der Segmentierung. Die Partikel sind annähernd perfekt von Modell R segmentiert worden. Modell S hingegen zeigt erneut einen zerfressenen Rand der Partikel. Dies wirkt sich auf die gemessene Größe der Partikel aus. Diese liegt bei Modell R bei 96 nm, was einer Abweichung von -11 % zur menschlichen Auswertung entspricht. Modell S weicht mit einer mittleren Kantenlänge von 92 nm um 15 % ab.

Die Probe zeigt deutlich, wie die verschiedenen Partikel einer Probe in unterschiedliche Klassen eingeteilt werden können. Für Proben, in denen die Partikelverteilung jeder Klasse einen eindeutigen Trend aufzeichnet, ist diese Art der Auswertung mittels KI erreichbar.

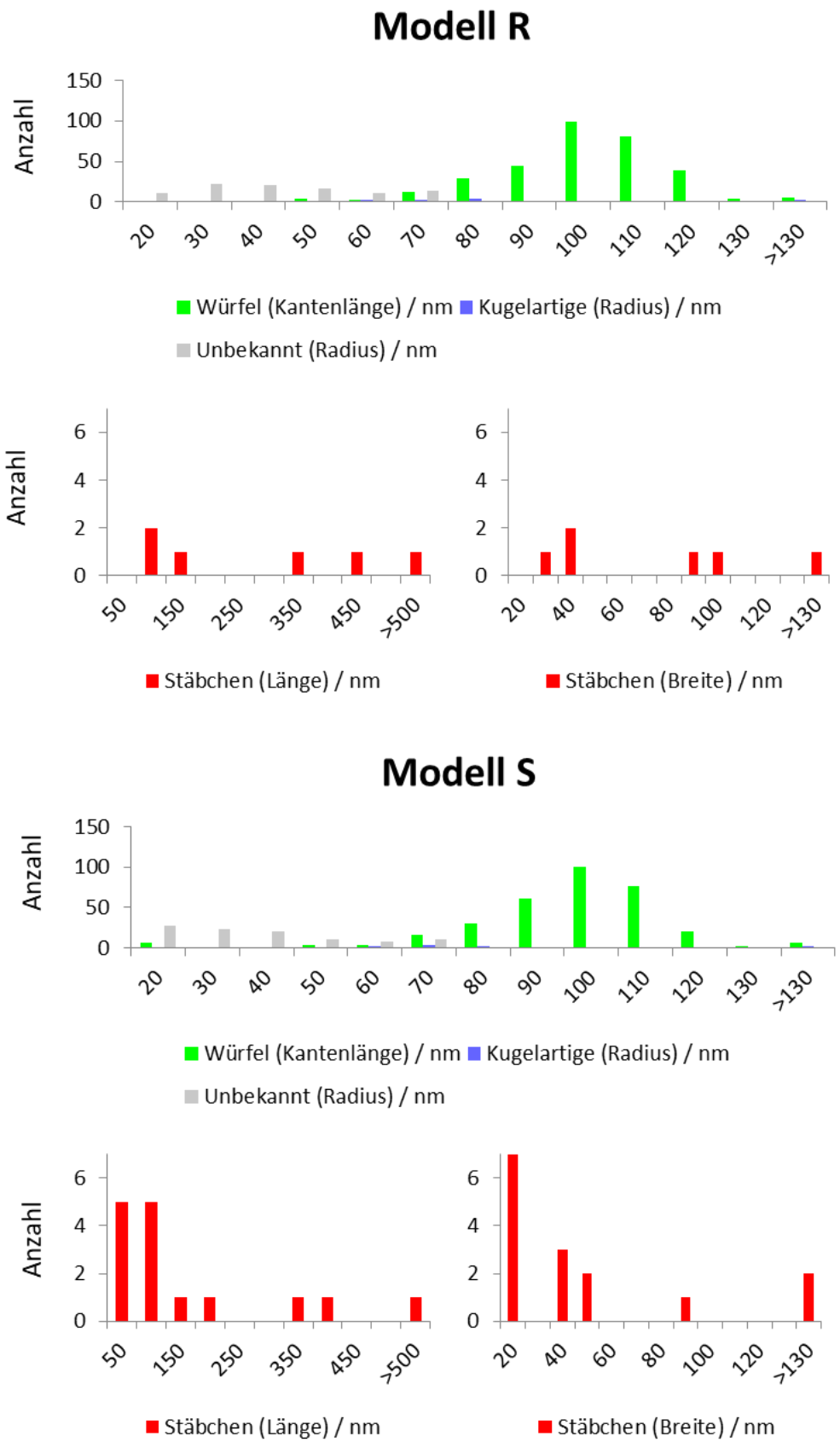


***Abbildung 72**: Partikelgrößenverteilung als Histogramme der Partikel, welche in **Abbildung 73** bei 50.000-facher Vergrößerung gezeigt sind.*

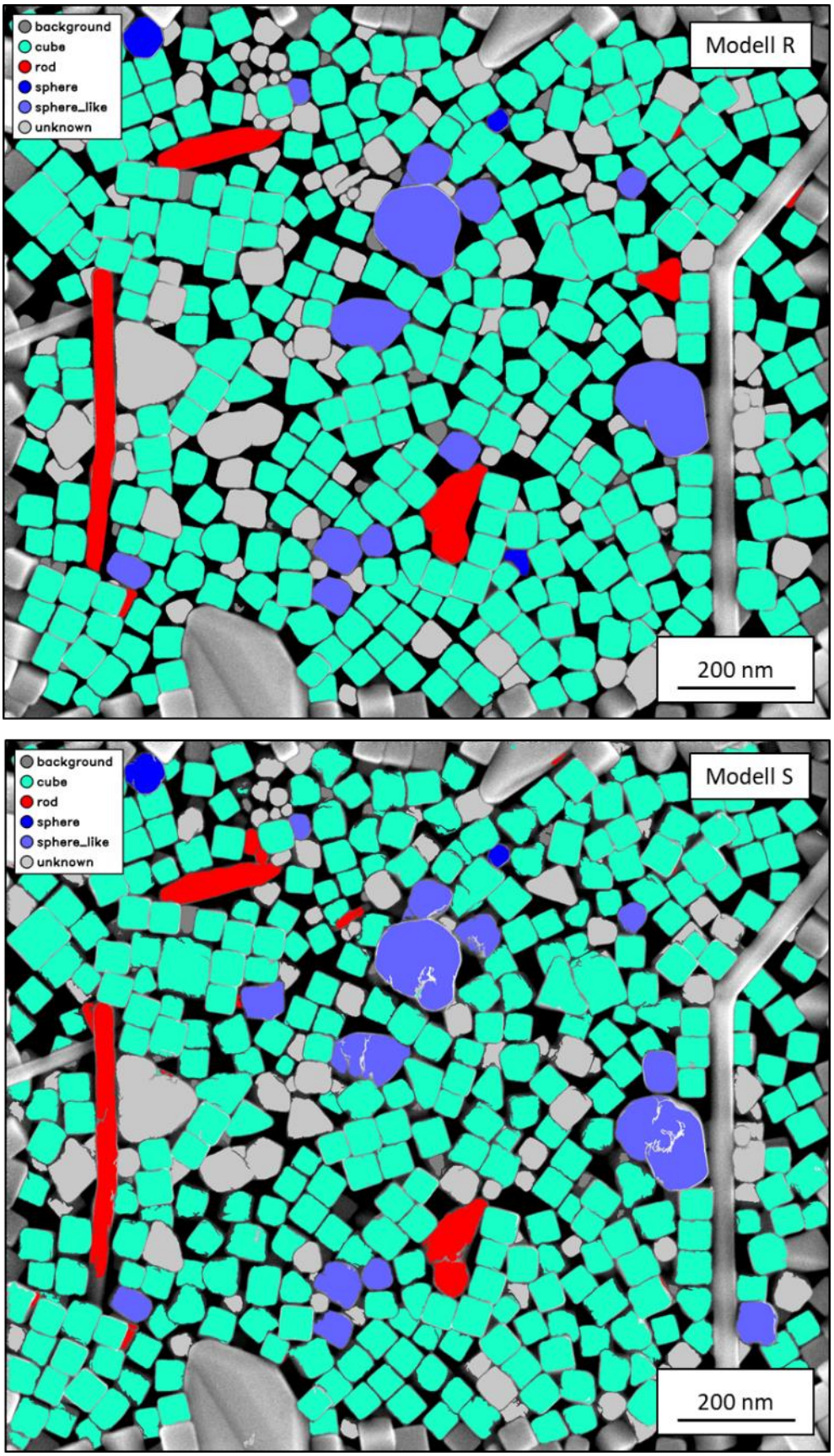


***Abbildung 73**: Automatische Auswertung von Ag Nanowürfeln. Die Skalierung beträgt 1,35 nm px-1.*

4.4.12 ZnO Mikrostäbchen bei 50.000-facher Vergrößerung (SE)

Die Auswertung einer reinen Probe von ZnO Mikrostäbchen ist in **Abbildung 75** gezeigt. Die Größenauswertung aller in der Abbildung vorhandenen Partikel ist in **Tabelle 18** gezeigt. Die Histogramme der Größenverteilung sind in Abbildung 74 dargestellt.

***Tabelle 18**: Auswertung der Partikelgrößen von ZnO Mikrostäbchen aus **Abbildung 75** sowie menschliche Auswertung der dominanten Partikelform.*

	Morphologie	**Ø / nm**	**n**
Mensch	Stäbchen (Länge)	400 ± 113	100
	Stäbchen (Breite)	100 ± 28	
	Länge / Breite	4	
Modell R	Stäbchen (Länge)	242 ± 119	807
	Stäbchen (Breite)	70 ± 36	
	Länge / Breite	3,59 ± 1,21	
Modell S	Stäbchen (Länge)	251 ± 136	707
	Stäbchen (Breite)	73 ± 44	
	Länge / Breite	3,61 ± 1,37	

Die Analyse der reinen Probe bestehend aus ZnO Mikrostäbchen, zeigt eines der Probleme der Segmentierung auf. Aufgrund der Länge von Stäbchen überlagern diese häufig, wodurch diese in mehrere Teilstücke segmentiert werden. Das Segmentierungsmodell konnte nicht eindeutig lernen, Stäbchen „ganz“ zu segmentieren. Dieser kritische Punkt wirkt sich auf die mittlere Stäbchenlänge der ausgewerteten Probe aus. Diese beträgt für Modell R 243 nm, wasnm, was einer Abweichung von -40 % zur menschlichen Auswertung entspricht. Modell S zeigt eine minimal geringere Abweichung. Die mittlere Länge beträgt hier 251 nm was einer Abweichung von -37 % zum Menschen entspricht. Die Abweichungen sind aufgrund des hohen Fehlers für Modell und Mensch nicht signifikant. Bei genauerer Betrachtung der Auswertung in **Abbildung 75** ist aber sichtbar, dass Partikel häufig von anderen Partikeln durchschnitten werden. Eine solche Auswertung ist daher nur eingeschränkt nutzbar.

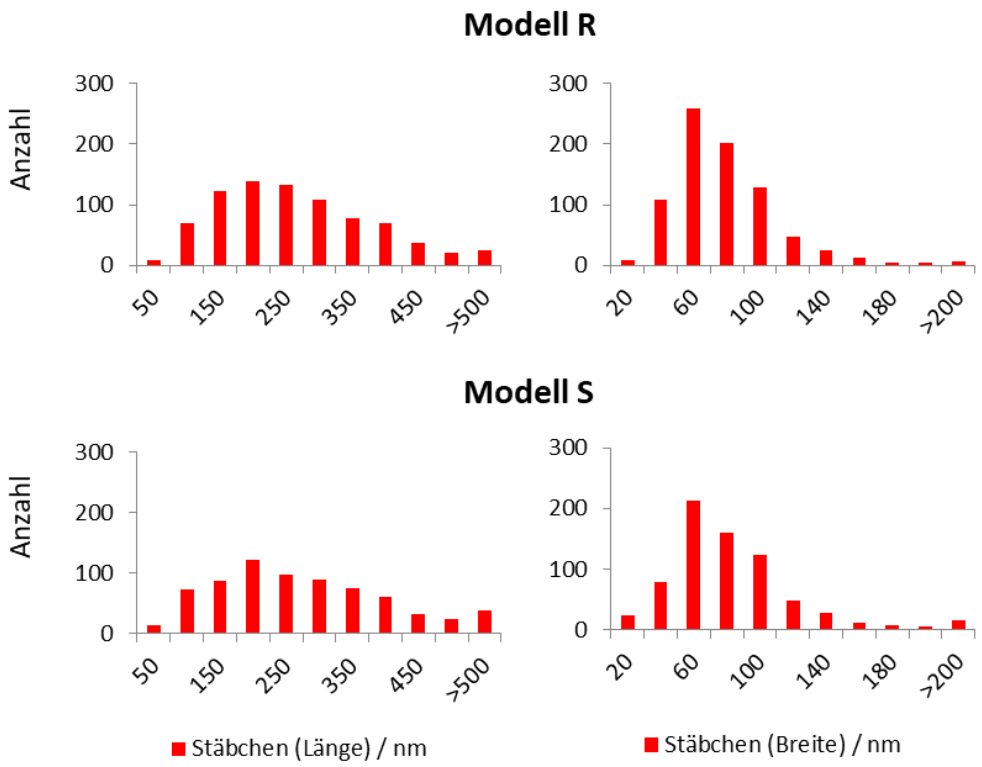


***Abbildung 74**: Partikelgrößenverteilung als Histogramme der Partikel, welche in **Abbildung 75** bei 50.000-facher Vergrößerung gezeigt sind.*

Die Klassifizierung der Partikel als Stäbchen erfolgte jedoch ohne größere Abweichungen. Partikel, die beispielsweise als kugelartiges Objekt klassifiziert wurden, zeigen bereits eine fehlerhafte Segmentierung durch verschmelzen von 2 oder mehr Stäbchen. Zumindest der Punkt der Klassifizierung ist dem Modell gelungen.

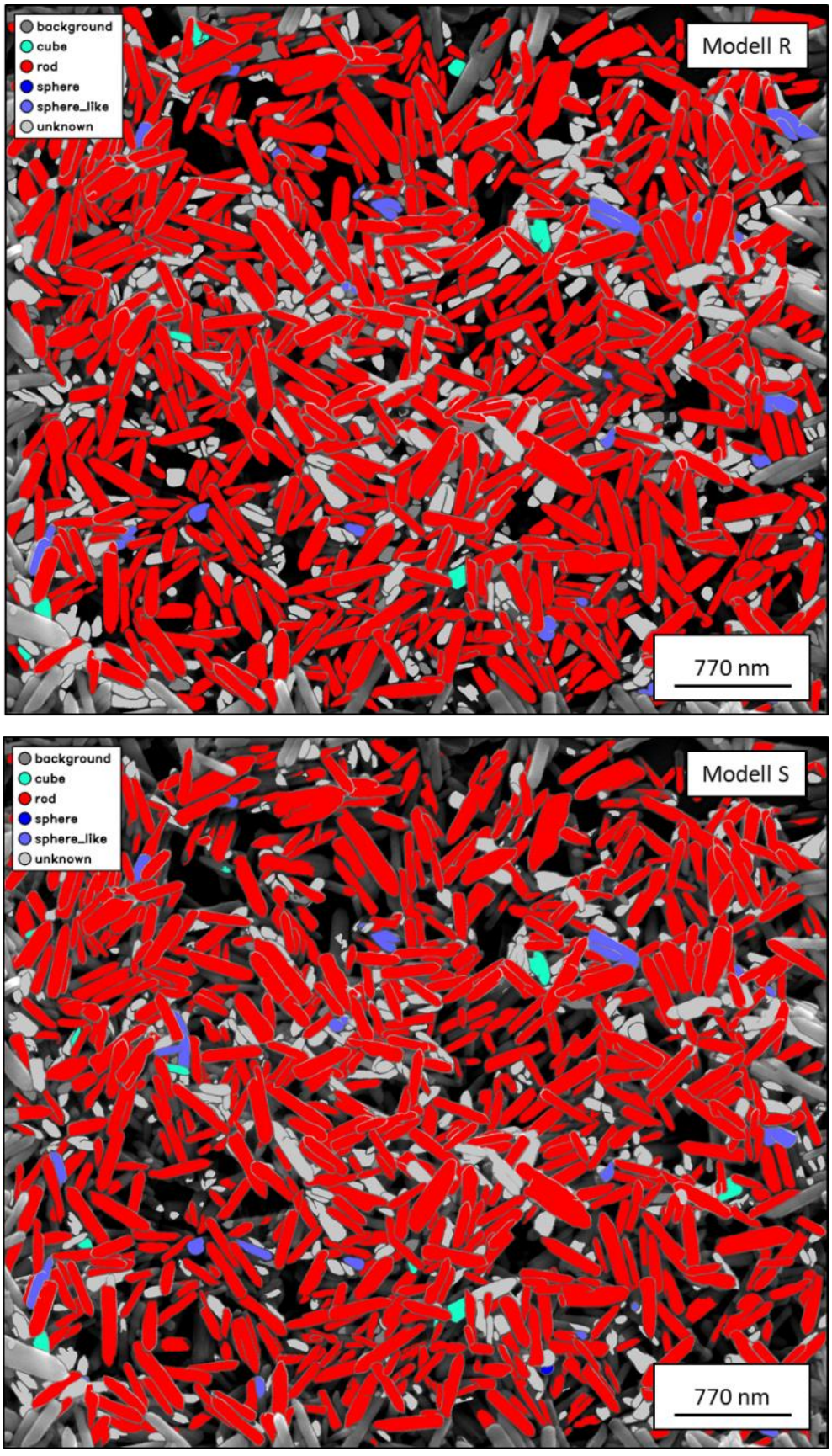


***Abbildung 75**: Automatische Auswertung von ZnO Mikrostäbchen. Die Skalierung beträgt 2,64 nm px-1.* [89]

4.4.13 TiO_2 Stäbchen bei 50.000-facher Vergrößerung (SE)

Die Auswertung einer reinen Probe von TiO_2 Mikrostäbchen ist in **Abbildung 77** gezeigt. Die Größenauswertung aller in der Abbildung vorhandenen Partikel ist in **Tabelle 19** gezeigt. Die Histogramme der Größenverteilung sind in **Abbildung 76** dargestellt.

***Tabelle 19**: Auswertung der Partikelgrößen von TiO_2 Stäbchen aus **Abbildung 77** sowie menschliche Auswertung der dominanten Partikelform. Durch die Überlappung diverser Partikel ist die Auswertung nicht repräsentativ. Daher wurde auf eine Darstellung des Seitenverhältnisses verzichtet.*

	Morphologie	Ør / nm	n
Mensch	Stäbchen (Länge)	1800 ± 180	ca. 50
	Stäbchen (Breite)	370 ± 65	
Modell R	Stäbchen (Länge)	1124 ± 696	148
	Stäbchen (Breite)	695 ± 340	
Modell S	Stäbchen (Länge)	1052 ± 694	166
	Stäbchen (Breite)	385 ± 286	

Die Analyse der Probe, bestehend aus TiO_2 Mikrostäbchen, zeigt ein weiteres Problem der Segmentierung auf. Die Probe enthielt eine Mischung aus stäbchenförmigen und sternförmigen Partikeln. Sternförmige Partikel traten in keinem der Datensätze auf. Zudem ähnelt die Basis der Sterne (Bereich, in dem die einzelnen Ausläufer zusammen laufen) ein ähnliches Erscheinungsbild wie der Übergang von 2 Partikel zueinander. Des Weiteren überlagern die Ausläufer / Strahlen der Sternpartikel in der Aufnahme. Diese Punkte führen zu einer gesplitterten Segmentierung einzelner Sternpartikel in kleinere und dementsprechend falsch segmentierte Partikel. Die Stäbchen hingegen werden größtenteils vollständig segmentiert. Weiterhin zeigt sich in der Segmentierung von Modell S eine erhebliche Menge falsch segmentierten Hintergrundes. Es ist unerklärlich, warum dieser dann vom Klassifizierungsmodell als eine der trainierten Formen erkannt wurde. Es ist möglich, dass die Strukturen des Probenträgers dies hervorgerufen haben.

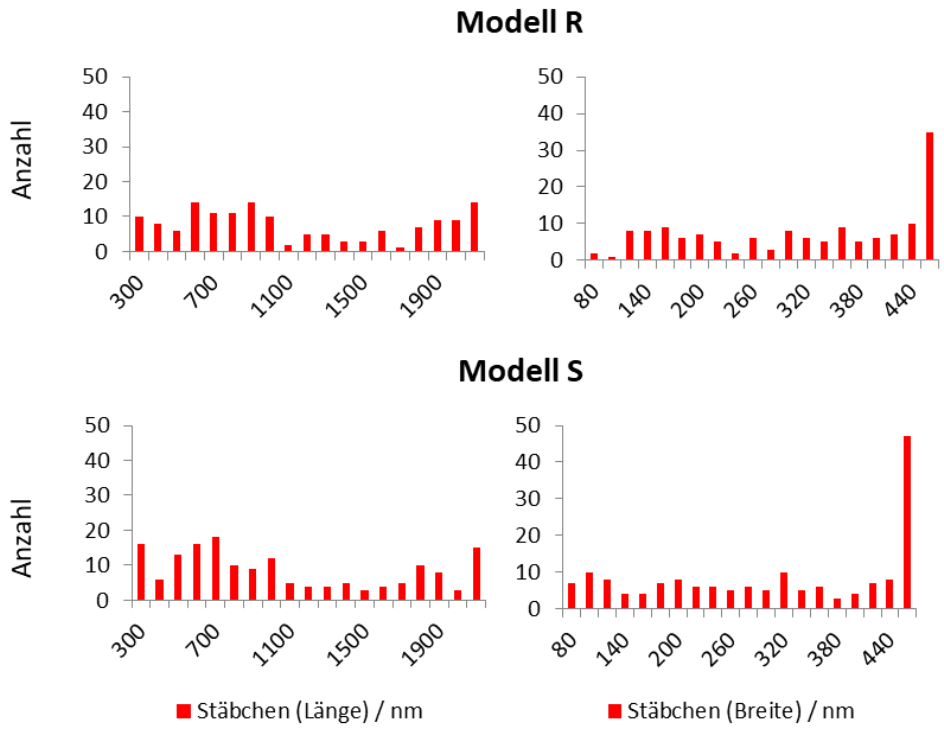


***Abbildung 76**: Partikelgrößenverteilung als Histogramme der Partikel, welche in **Abbildung 77** bei 50.000-facher Vergrößerung gezeigt sind.*

Fälschlicherweise gesplitterte Partikel wurden wie auch vollständige Partikel als Stäbchen identifiziert. Die führt zu einer Verzerrung der Größenverteilung. Eine mittlere Größe wahrer Stäbchen lässt sich somit nicht ermitteln. Erneut zeigt sich die mangelnde Leistung des Workflows bezüglich der Segmentierung und Klassifizierung von Stäbchen. Darüber hinaus zeigt es, dass sternförmige Partikel nicht erkannt werden können. Dies lässt vermuten, dass auch andere exotische Partikelformen wie planare Objekte oder hohle Partikel nicht vollständig segmentiert werden können. Besonders Applikationen, in denen Partikel mit einem für Nanopartikel hohen Oberfläche zu Volumen Verhältnis angestrebt werden, kann die hier gezeigte Methode nicht unterstützend angewendet werden.

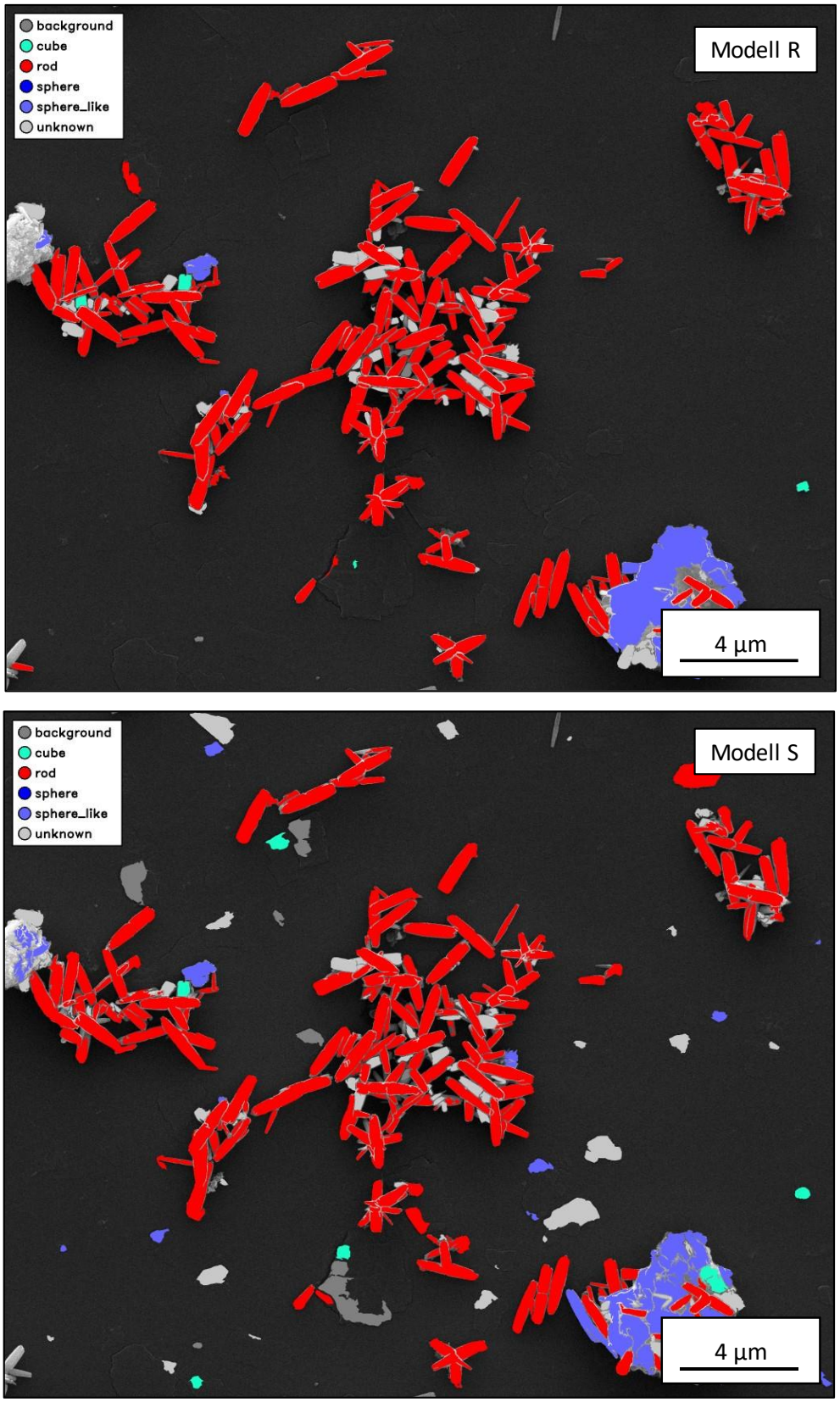


***Abbildung 77**: Automatische Auswertung von TiO_2 Mikropartikel. Die Skalierung beträgt 26,4 nm px-1.* [89]

Aufgrund des hohen Fehlers ist eine verlässliche Größenbestimmung aller im Bild enthaltenen Partikel nicht möglich. Die Ergebnisse des Workflows beinhalten eine vollständige Auswertung zu jedem einzelnen Partikel, wodurch es dem menschlichen Analysten möglich ist, einzelne vollständig und korrekt segmentierte Partikel in dem Bild zu identifizieren und mit der Partikeleigenen Nummerierung abzugleichen, um die von der KI bestimmte Länge und Breite zu erhalten. Der damit verbundene Arbeitsaufwand ist jedoch höher als die manuelle Auswertung, womit die Auswertung mittels KI obsolet wird. Es zeigt sich in den vorangegangenen Bildern von Stäbchen, dass diese nur vermindert auswertbar sind.

5 Zusammenfassung

Die hier vorliegende Arbeit umfasste mehrere Teilbereiche des maschinellen Lernen und kombinierte diese zu einem finalen Workflow zur Unterstützung eines menschlichen Analysten für die Auswertung von elektronenmikroskopischen Aufnahmen. Es wurden zunächst mehrere Datensätze aus REM-Bildern angefertigt, um die entsprechenden Modelle zu trainieren. Die Modelle wurden aus einer Anzahl bisher in der Literatur veröffentlichter Architekturen ausgewählt, in ihrer publizierten Form getestet und anschließend für die in dieser Arbeit nötigen Aufgaben angepasst. Weiterhin wurden neuronale Netze genutzt, um künstliche REM-Bilder anzufertigen, die im optimalen Falle menschlich markierte Bilder oder Klassifizierungen einzelner Bilder obsolet gemacht hätten.

Der erste Schritt bestand aus der Erstellung eines Datensatzes für die Segmentierung von STEM-Bildern. Für diesen wurden echte STEM-Bilder von verschiedenen Partikelsystemen aufgenommen und zusätzlich aus einer Vielzahl von gespeicherten Aufnahmen ausgewählt. Zu jedem Bild wurde händisch eine Annotationsmaske erstellt, die das Bild binär in Vordergrund (Partikel) und Hintergrund (Probenträger) teilt. Mit Bildern und Annotationsmasken konnte ein Modell trainiert werden, das diesen Schritt automatisiert. Das Modell war eine Weiterentwicklung der erprobten UNet Architektur. Es wurden für jedes Bild Gewichtungsmatrizen berechnet, welche den Hintergrundpixeln in Abhängigkeit ihrer Entfernung zu den nächsten Partikeln einen Wert zuwiesen. Je höher der assoziierte Wert dieses Pixels, desto stärker war die "Aufmerksamkeit“ des Modells während des Trainings auf diesem Bereich. Hintergrundpixel des Bildes, die zwischen Partikeln lagen, erhielten eine hohe Gewichtung, während Hintergrundpixel, die keine Partikelgrenzen in näherer Umgebung sahen, keine Gewichtung erhielten. Da diese Bereiche besonders wichtig sind, um nahe liegende Partikel zu trennen, lernte das Modell einen starken Fokus auf diese Bereiche zu lenken. Um eine gleichbleibend hohe Leistung sicherzustellen, wurden diese Gewichtungsmatrizen weiterentwickelt. Zusätzlich zur Entfernung der Pixel zu jedem Partikel des Bildes wurde die Intensität der Pixel selbst in den Gewichtungskarten berücksichtigt. Durch diesen zusätzlichen Fokus auf helle Bereiche zwischen 2 Partikeln konnten auch schwer trennbare Partikel getrennt werden. Erst durch die Trennung einzelner Partikel wurde die gewünschte Leistung erreicht. Das Modell erreichte eine hohe Leistung in der Segmentierung individueller Nanopartikel. Einzelne Nanopartikel konnten somit identifiziert werden.

Darauffolgend wurde ein Modell für die Klassifizierung einzelner Nanopartikel entwickelt. Für dieses Modell wurde die bereits erprobe AlexNet Architektur weiter modifiziert. Die Modifizierung erfolgte nach Betrachtung der Auslastung einzelner Neuronen der vollvernetzten Schichten. Nachdem validiert werden konnte, dass diese zu großen Teilen unbeansprucht sind, wurden die Neuronen reduziert. Dies erfolgte vor allem unter Berücksichtigung der durchschnittlichen verfügbaren Computerleistung, da davon ausgegangen werden musste, dass nicht jeder PC eine Grafikkarte besitzt und die Geschwindigkeit der automatischen Analyse hoch sein muss. Für die Erstellung des Datensatzes wurden mehrere 100.000 einzelne Nanopartikel aus REM-Bildern ausgeschnitten. Diese wurden dann manuell in die in dieser Arbeit aufgezeigten Klassen für STEM-Partikel eingeteilt.

Aufgrund der diversen Erscheinung von STEM- und SE-Bildern musste für die Analyse von SE-Bildern ein weiteres KI-System trainiert werden. Dies begann erneut mit der Erstellung eines Datensatzes für das Training eines UNet abgeleiteten Modells. Der Datensatz bestand aus SE-Bildern und den manuell erstellten binären Annotationsmasken. Nachdem die Annotationsmasken erstellt waren, konnten auch für SE-Bilder Gewichtungskarten berechnet werden. Mittels echter REM-Bilder, den manuellen Annotationsmasken und den Gewichtungskarten wurde das UNet Modell trainiert. Es ist in der Lage, diverse Partikelformen akkurat voneinander zu trennen. Einzig stäbchenförmige Partikel lassen sich nicht vollständig identifizieren. Auch hier wurde im Anschluss die Klassifizierung von einem weiteren Netzwerk übernommen. Hierfür wurde ResNet34 verwendet, welches aufgrund seiner Struktur sehr leistungsstark ist. Um ResNet zu trainieren, wurden erneut mehrere 100.000 Partikel aus REM-Bildern ausgeschnitten und manuell in die in dieser Arbeit verwendeten Klassen eingeteilt. ResNet erreichte daraufhin eine überragende Performance in der Erkennung einzelner Klassen.

Parallel zu der Erstellung der Datensätze und dem Training der neuronalen Netze wurden Umfragen bezüglich der Sicherheit des Menschen in der Erkennung von Nanopartikeln in REM-Bildern durchgeführt. Dies führte zur Erkenntnis folgender Problematik: Eine Gruppe von Menschen ist sich oft unsicher, zu welcher Morphologie ein gezeigtes Partikel gehört. Das heißt, die von Menschen gemachten Daten werden von einer unabhängigen Gruppe von Menschen mit hoher Wahrscheinlichkeit als fehlerhaft abgetan werden. Dies resultierte in der Erschaffung von künstlichen REM-Bildern mittels eines weiteren Netzwerktyps. Für die

Erstellung von künstlichen REM-Bildern wurde ein zyklisches generatives neuronales Netz verwendet (CycleGAN). Dieses lernte aus zuvor simulierten Höhenkarten und echten REM-Bildern das REM typische Aussehen von REM-Bildern auf die simulierten Höhenkarten zu übertragen. Dadurch wurde es möglich, diverse Partikelformen zunächst nach definierten Maßstäben zu designen und anschließend in einer Simulation auf den Probenträger fallen zu lassen. Die genutzte Software Blender war nicht nur in der Lage, die Partikel zu generieren und die Simulationen durchzuführen, sondern auch fehlerfreie künstliche Annotationsmasken der Partikel zu exportieren.

Somit war es möglich, ein Segmentierungsmodell vollständig auf künstliche Daten zu trainieren. Zunächst wurde eine Anzahl von simulierten Höhenkarten erstellt. Zu jeder Höhenkarte existierte eine fehlerfreie Annotationsmaske. Die Höhenkarten wurden mittels eines trainierten GANs in synthetische REM-Bilder umgewandelt. Synthetische REM-Bilder und simulierte Annotationsmasken wurden verwendet, um ein Segmentierungsmodell nach gleichem Schema wie für SE-Bilder zu trainieren. Das so trainierte Modell wurde dann mittels gleichem echten Datensatz von SE-Bildern validiert wie auch das reguläre auf reale Daten trainierte Modell. Die Validierung des so trainierten Modells offenbarten Schwachpunkte, die in dieser Arbeit nicht mehr angegangen wurden.

Abschließend wurden die trainierten Modelle zu einem Workflow zusammengefügt. Dieser Workflow bestand aus je einem Modell zur Segmentierung von SE- und STEM-Bildern und je ein Modell zur Klassifizierung von SE- und STEM-Bildern. Dadurch wurde die Möglichkeit geschaffen, einzelne Bilder mittels weniger Einstellungen vollständig auszuwerten. Die Auswertung eines Bildes mit mehreren Tausend Nanopartikeln dauert weniger als 5 min.

Diese Technologie bietet die Möglichkeit, automatische Analysen von diversen Proben auszuführen. Die verwendeten Modelle wurden bereits vielfach genutzt und zeigen auch in anderen Bereichen wie der direkten Objektlokalisierung großen Erfolg. UNet, ResNet und VGG sind Architekturen, die trotz der Entwicklung neuerer Modelle nicht an Bedeutung verloren haben. Besonders ResNet sticht hierbei hervor. Die Identitäts-Schichten, die der Architektur so viel Leistung in der Bild-Analyse geben, werden auch in Modellen wie Alpha-Go und GPT-3 eingesetzt. Als dediziertes Netzwerk für die Segmentierung von medizinischen Bildern hat UNet bewiesen, dass es akkurater arbeitet als zuvor bekannte Architekturen. Da besonders in medizinischen Aufnahmen eine auf den Pixel genaue Analytik wichtig ist und oftmals geringe

Abweichungen in der Segmentierung zwischen Detektion und Missachtung von Krankheiten eine Rolle spielen, ist UNet eine herausragende Architektur. Wie bereits eingangs beschrieben, zeigen bisherige Publikationen einen eklatanten Mangel von publizierten Datensätzen. Die publizierten Modelle sind auf kleine und zudem sehr invariabel gehaltene Datensätze trainiert worden. Ein allgemeiner und vielfach geprüfter Datensatz existiert nicht. Für diese Arbeit wurde ein Datensatz geschaffen, der größer war als alle bisher gezeigten Datensätze im Nanobereich. Die Leistung der so trainierten Modelle konnte vielfach unter verschiedenen Bedingungen gezeigt werden. Die Applikation dieser modernen Netzwerke in der Erforschung von Nanomaterialen ist somit ein Schritt in die Einbeziehung modernster computergestützter Analyseverfahren. Die Arbeit zeigt die Einbindung von bekannten KI-Verfahren in die Analytik von REM-Bildern im Nanobereich und eine Weiterentwicklung zu diesem Feld.

Weiterhin bleiben auch Fragen offen, die möglicherweise zu einer besseren Leistung als auch Schnelligkeit des Systems führen könnten. Zunächst könnte die Klassifizierung von Nanopartikeln mittels Transfer-Learning vereinfacht werden. Anstatt, wie hier gezeigt, würde ein vortrainiertes Netzwerk vom ResNet-Typ genutzt werden, um die Eigenschaften der Nanopartikel zu identifizieren. Für SE- und STEM-Bilder würde ein individueller Klassifizierungskopf trainiert werden. Somit würde die Anzahl der benötigten Parameter um ca. 50 % reduziert werden. Weiterhin bleibt ein Großteil der Parameter des Netzwerkes eingefroren. Letztendlich würden nur wenige Tausend Parameter trainiert werden müssen. Dieses Training ist auch auf kleineren Computern schnell auszuführen. Dadurch ergebe sich die Möglichkeit, die Datensätze zu erweitern und die Modelle auf neue Nanopartikelformen auszubauen.

Künstliche Datensätze, wie hier gezeigt bieten eine weitere Möglichkeit der Verbesserung. Das verwendete GAN zeigte die Fähigkeit, künstliche Bilder zu generieren. Aufgrund der Zusammensetzung des GANs ergibt sich die Möglichkeit, 2 andere Netzwerk-Architekturen einzusetzen. Diskriminator und Generator können individuell optimiert werden, um realistischere Bilder zu erzeugen: (i) Die Analyse mittels VGG konnte zeigen, dass die Bilder nicht variabel sind. Eine Erweiterung der verwendeten Generatoren durch das Hinzufügen weiterer Identitätsmodule würde die Anzahl der erlernbaren Parameter deutlich erhöhen. Hierdurch ergibt die Problematik eines insgesamt schwierigeren Trainings als auch die Perspektive realistischerer Bilder zu erzeugen. Ein vollständiger Austausch der Netzwerke

hingegen ist weniger ratsam, da die verwendeten ResNet-Generatoren als die leistungsstärksten Netzwerke gelten. (ii) Die Diskriminatoren bieten weitere Optionen für die Optimierung. Zum einen könnten hier auch weitere Schichten eingefügt werden, um das rezeptive Feld zu vergrößern. Somit „sieht“ der Diskriminator größere Bildbereiche. Die Klassifizierung und somit das gesamte Training könnte stabilisiert werden. Zusätzlich könnten auch andere Verlustfunktionen eingesetzt werden. Dies würde die Netzwerke zwingen, sich auf andere Dinge zu „konzentrieren“.

Insgesamt bietet die in dieser Arbeit vorgestellte Thematik viel Raum für weitere Exploration. Die aufgezeigten Möglichkeiten zur Verbesserung sind nur einige konkrete Schritte, die in besseren Resultaten enden könnten. Besonders die Entwicklung fehlerfreier Datensätze ohne den Einfluss des Menschen bieten eine spannende Herausforderung für die Zukunft.

6 English Summary

The work here involved several sub-areas of machine learning and combined these into a final workflow to support a human analyst for the analysis of scanning electron micrographs. First, several datasets were made from SEM images in order to train different models. The models were selected from a number of architectures previously published in the literature, which were tested in their naïve form and then adapted for the tasks required in this work. Furthermore, neural networks were used to create artificial SEM images, which ideally would have made human-labelled images or classifications of individual images obsolete.

The first step consisted of creating a dataset for the segmentation of STEM-images. For this, real STEM-images of different particle systems were recorded and additionally selected from a large number of saved images. An annotation mask was created manually for each image, which binary representation of the image, splitting it into foreground (particles) and background (sample holder). With images and annotation masks, a model could be trained that automates this step. The model was a development of the previously published UNet architecture. Weighting matrices, called weight maps, were calculated for each image, which assigned a value to the background pixels depending on their distance to the nearest particle borders. The higher the associated value of that pixel, the stronger the model's "attention" to that region during training. Background pixels of the image that lay between particles received a higher weight, while background pixels that did not experience particle boundaries nearby received no weight. Since these areas are particularly important for separating nearby particles, the model learned to place a strong focus on these areas. To ensure consistently high performance, these weight matrices were further developed. In addition to the distance of background pixels to the particles of the image the intensity of the pixels themselves was considered for the weight maps. This additional focus on bright areas between 2 particles also made it possible to separate difficult-to-separate particles. Only by separating individual particles the desired performance was achieved. The model achieved high performance in segmenting individual nanoparticles. Individual nanoparticles could thus be identified with another model.

Subsequently, a model for the classification of individual nanoparticles was developed. The already tested AlexNet architecture was further modified for this model. The modification was made after considering individual Neurons in the fully connected layers. After it could be

validated that these are largely empty in, meaning no signal coming out of these, the Neurons in these layers were reduced. This was mainly done because of the average available computer performance, since it had to be assumed that not every PC is equipped with a graphics card and the speed of the automatic analysis must be high. To create the data set, several 100,000 individual nanoparticles were cut out of SEM images. These were then manually divided into the classes for STEM-particles shown in this work.

Due to the diverse appearance of STEM- and SE-images, another AI system had to be trained for the analysis of SE-images. This started with recreating a data set for training a UNet transmitted model. The data set consists of SE-images and the manually created binary annotation masks. After the annotation masks were created, weight maps have been calculated for the SE-dataset. The UNet model was trained using real SEM images, the manual annotation masks and the weight maps. It is able to accurately separate various particle shapes from each other. Only rod-shaped particles cannot be fully identified. Here, too, the classification was then performed by another network. Therefore, ResNet34 was used, which is very powerful due to its architecture. To train ResNet, several 100,000 particles were cut out from SEM images and manually classified into the classes used in this work. ResNet then achieved outstanding performance in recognizing individual nanoparticles.

Parallel to the creation of the datasets and the training of the neural networks, surveys were carried out regarding the confidence of humans in recognizing nanoparticles in SEM images. This led to the realization of the following problem: A group of people is often unsure to which morphology a particle shown belongs. That means, the human-made data is likely to be classified as erroneous by an independent group of people. This resulted in the creation of artificial SEM images using another type of network. A cyclic generative neural network (CycleGAN) was used to create artificial SEM images. This learned from previously simulated height maps and real SEM images to transfer the typical appearance of SEM images to the simulated height maps. This made it possible to first design various particle shapes according to defined needs and then to drop them onto the sample holder in a simulation. The free software Blender was used not only able to generate the particles and carry out the simulations, but also to export error-free artificial annotation masks of the scenery.

Thus, it was possible to train a segmentation model completely on artificial data. First, a number of simulated height maps were created. An error-free annotation mask existed for

each height map. The height maps were converted into synthetic SEM images using a trained CycleGAN. Synthetic SEM images and simulated annotation masks were used to train a segmentation model using the same training procedure as for SE-images. The model trained in this way was then validated using the same real data set of SE-images as the regular model trained on real data. The validation of the model trained in this way revealed weak points that were not addressed in this work.

Finally, the trained models were merged into a workflow. This workflow consisted of one model each for segmenting SE- and STEM-images and one model each for classifying SE- and STEM-images. This made it possible to fully evaluate individual images. The evaluation of an image with several thousand nanoparticles takes less than 5 minutes.

This technology offers the possibility to carry out automatic analysis of various samples. The adapted models have already been used widely and have also shown great success in other areas such as object localization. UNet, ResNet and VGG are architectures that have not lost their importance despite the development of newer models. ResNet in particular stands out here. The identity layers that give the architecture so much power in image analysis are also used in models like Alpha-Go and GPT-3. As a dedicated network for segmenting medical images, UNet has proven to perform more accurately than previously known architectures. Because pixel-precise analytics is particularly important in medical images, UNet is an outstanding architecture. As already described at the beginning, previous publications show a lack of published data sets. The published models have been trained on small and invariable datasets. A general and frequently updated dataset does not exist. For this work, a dataset was created that was larger than any previously shown dataset of nanoparticles. The performance of the models trained on this dataset have been shown for several cases under different conditions. The application of these modern networks in the research of nanomaterials is thus a step in the inclusion of the most modern computer-aided analysis methods. The work shows the integration of well-known AI-methods in the analysis of SEM images in the nano scale and a further development in this field.

Furthermore, questions remain unanswered that could possibly lead to better performance and speed of the system. First, the classification of nanoparticles could be simplified using transfer learning. Instead, as shown here, a pre-trained ResNet-type network would be used to identify the features of the nanoparticles. An individual classification head would be trained

for SE- and STEM-images. This would reduce the number of parameters required by around 50 %. Furthermore, a large number of parameters of the network remain frozen. Ultimately, only a few thousand parameters would need to be trained. This training can also be carried out quickly on smaller computers. This results in the possibility of expanding the datasets and expanding the models to new nanoparticle shapes continuously.

Artificial datasets, as shown here, offer another opportunity for improvement. The used GAN proofed the ability to generate artificial images. Due to the composition of the GAN, there is the possibility of using 2 other network architectures. The discriminator and generator can be individually optimized to produce more realistic images: (i) Analysis using VGG could show that the images are not variable. Expanding the generators used by adding more identity-modules would significantly increase the number of parameters that can be learned. This results in the problem of an overall more demanding training as well as creating the perspective of realistic images. A complete replacement of the networks, on the other hand, is less advisable, since the used ResNet-based generators are considered to be the most powerful networks. (ii) The discriminators provide further options for optimization. On the one hand, further layers could also be inserted here in order to enlarge the receptive field. The discriminator thus "sees" larger image areas. The classification and therefore the entire training could be improved. In addition, other loss functions could also be used. This would force the networks to "focus" more on “realness”.

Overall, the topic presented in this work offers a lot of room for further exploration. The identified improvement opportunities are just a few concrete steps that could lead to better results. In particular, the development of error-free datasets without human bias offers an exciting challenge for the future.

7 Literaturverzeichnis

1. Ma, H., Chen, Z., Gao, X., Liu, W., und Zhu, H. (2019). *3D hierarchically gold-nanoparticle-decorated porous carbon for high-performance supercapacitors*. Scientific Reports. 9. 17065

2. Jain, R., Lakhnot, A.S., Bhimani, K., Sharma, S., Mahajani, V., Panchal, R.A., Kamble, M., Han, F., Wang, C., und Koratkar, N. (2022). *Nanostructuring versus microstructuring in battery electrodes*. Nature Reviews Materials. 7. 736-746

3. Mitchell, M.J., Billingsley, M.M., Haley, R.M., Wechsler, M.E., Peppas, N.A., und Langer, R. (2021). *Engineering precision nanoparticles for drug delivery*. Nature Reviews Drug Discovery. 20. 101-124

4. Zelepukin, I.V., Griaznova, O.Y., Shevchenko, K.G., Ivanov, A.V., Baidyuk, E.V., Serejnikova, N.B., Volovetskiy, A.B., Deyev, S.M., and Zvyagin, A.V. (2022). *Flash drug release from nanoparticles accumulated in the targeted blood vessels facilitates the tumour treatment*. Nature Communications. 13. 6910

5. Hughes, A.S., Liu, Z., Raftari, M., und Reeves, M.M. (2015). *A workflow for characterizing nanoparticle monolayers for biosensors: Machine learning on real und artificial SEM images*. PeerJ PrePrints. 2. e671v671

6. Sommer, C., Straehle, C., Köthe, U., und Hamprecht, F.A. (2011). *ilastik: Interactive Learning und Segmentation Toolkit*. In Proceedings of the ISBI 2011. 230-233

7. Otsu, N. (1979). *A Threshold Selection Method from Gray-Level Histograms*. IEEE Transactions on Systems, Man, und Cybernetics. 9. 62-66

8. Jara-Lugo, L.A., Caro-Gutiérrcz, J., González-Navarro, F.F., Curiel-Álvarez, M.A., Pérez-Landeros, Ó.M., und Radnev-Nedev, N. (2022). *Nanoparticles diameter characterization using image analysis methodology*. In Proceedings of the IEEE Mexican International Conference on Computer Science 2022. 1-7

9. Muneesawang, P. und Sirisathitkul, C. (2015). *Size Measurement of Nanoparticle Assembly Using Multilevel Segmented TEM Images*. Journal of Nanomaterials. 16. 1687-4110

10. De Siqueira, A.F., Cabrera, F.C., Pagamisse, A., und Job, A.E. (2014). *Segmentation of scanning electron microscopy images from natural rubber samples with gold nanoparticles using starlet wavelets*. Microscopy research and technique. 77. 71-78

11. Baiyasi, R., Gallagher, M.J., McCarthy, L.A., Searles, E.K., Zhang, Q., Link, S., und Landes, C.F. (2020). *Quantitative Analysis of Nanorod Aggregation and Morphology from Scanning Electron Micrographs Using SEMseg*. The Journal of Physical Chemistry A. 124. 5262-5270

12. Berg, S., Kutra, D., Kroeger, T., Straehle, C.N., Kausler, B.X., Haubold, C., Schiegg, M., Ales, J., Beier, T., Rudy, M., Eren, K., Cervantes, J.I., Xu, B., Beuttenmueller, F., Wolny, A., Zhang, C., Koethe, U., Hamprecht, F.A., und Kreshuk, A. (2019). *ilastik: interactive machine learning for (bio)image analysis*. Nature Methods. 16. 1226-1232

13. Ren, S., He, K., Girshick, R., und Sun, J. (2017). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. IEEE Transactions on Pattern Analysis und Machine Intelligence. 39. 1137-1149

14. Monchot, P., Coquelin, L., Guerroudj, K., Feltin, N., Delvallée, A., Crouzier, L., und Fischer, N. (2021). *Deep Learning Based Instance Segmentation of Titanium Dioxide Particles in the Form of Agglomerates in Scanning Electron Microscopy*. Nanomaterials. 11. 968

15. Kim, H., Han, J., und Han, T.Y.-J. (2020). *Machine vision-driven automatic recognition of particle size and morphology in SEM images*. Nanoscale. 12. 19461-19469

16. Holm, E.A., Cohn, R., Gao, N., Kitahara, A.R., Matson, T.P., Lei, B., und Yarasi, S.R. (2020). *Overview: Computer Vision and Machine Learning for Microstructural Characterization and Analysis*. Metallurgical und Materials Transactions A. 51. 5985-5999

17. Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., Park, C.W., Choudhary, A., Agrawal, A., Billinge, S.J.L., Holm, E., Ong, S.P., und Wolverton, C. (2022). *Recent advances and applications of deep learning methods in materials science*. Computational Materials. 8. 59

18. Simonyan, K. und Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. ArXiv preprint at arxiv:1409.1556.

19. He, K., Zhang, X., Ren, S., und Sun, J. (2016). *Deep residual learning for image recognition*. In Proceedings of the IEEE CVPR 2016. 770-778

20. Wang, X., Li, J., Ha, H.D., Dahl, J.C., Ondry, J.C., Moreno-Hernandez, I., Head-Gordon, T., und Alivisatos, A.P. (2021). *AutoDetect-mNP: An Unsupervised Machine Learning Algorithm for Automated Analysis of Transmission Electron Microscope Images of Metal Nanoparticles*. Journal of the American Chemical Society. 1. 316-327

21. DeCost, B.L. and Holm, E.A. (2016). *A large dataset of synthetic SEM images of powder materials and their ground truth 3D structures*. Data in brief. 9. 727-731

22. LeCun, Y., Boser, B., Denker, J.S., Howard, R.E., Habbard, W., Jackel, L.D., und Henderson, D. (1990). *Handwritten digit recognition with a back-propagation network*. In: Advances in neural information processing systems 2. 1. Auflage. MIT Press. 396–404

23. Szegedy, C., Wei, L., Yangqing, J., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., und Rabinovich, A. (2015). *Going deeper with convolutions*. In Proceedings of the IEEE CVPR 2015. 1-9

24. Krizhevsky, A., Sutskever, I., und Hinton, G.E. (2012). *ImageNet classification with deep convolutional neural networks*. In Proceedings of the NIPS 2012. 1097–1105

25. Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. 2. Auflage. MIT Press

26. Gatys, L.A., Ecker, A.S., und Bethge, M. (2016). *Image Style Transfer Using Convolutional Neural Networks*. In Proceedings of the IEEE CVPR 2016. 2414-2423

27. Zeiler, M.D. und Fergus, R. (2014). *Visualizing and Understanding Convolutional Networks*. In Proceedings of the Computer Vision – ECCV 2014. 818-833

28. Lee, H., Grosse, R., Ranganath, R., und Ng, A. (2011). *Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks*. Communications of the Association for Computing Machinery. 54. 95-103

29. Tesla Inc. (2023). https://www.tesla.com/de_de. Letzmalig Besucht am 19.01.2023

30. Rosenblatt, F. (1958). *The perceptron: a probabilistic model for information storage and organization in the brain*. Psychological review. 65. 386

31. Widrow, B. und Hoff, M.E. (1962). *Associative Storage und Retrieval of Digital Information in Networks of Adaptive "Neurons"*. In: Biological Prototypes und Synthetic Systems. 1. Auflage. 160-160

32. Jain, A.K., Jianchang, M., and Mohiuddin, K.M. (1996). *Artificial neural networks: a tutorial*. Computer. 29. 31-44

33. Hebb, D.O. (1949). *The organization of behavior; a neuropsychological theory*. 1. Auflage. Taylor & Francis Inc

34. Rumelhart, D.E., Hinton, G.E., und Williams, R.J. (1986). *Learning representations by back-propagating errors*. Nature. 323. 533-536

35. Fukushima, K. (1980). *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*. Biological Cybernetics. 36. 193-202

36. Venkatesh, S. und Owens, R. (1990). *On the classification of image features*. Pattern Recognition Letters. 11. 339-349

37. Canny, J.F. (1983). *Finding edges und lines in images*. Master Thesis - MIT

38. Hassaballah, M., Abdelmgeid, A.A., and Alshazly, H.A. (2016). *Image Features Detection, Description und Matching*. In: Image Feature Detectors and Descriptors : Foundations and Applications. 1. Auflage. Springer. 11-45

39. LeCun, Y., Bengio, Y., und Hinton, G. (2015). *Deep learning*. Nature. 521. 436-444

40. Fissan, H., Ristig, S., Kaminski, H., Asbach, C., und Epple, M. (2014). *Comparison of different characterization methods for nanoparticle dispersions before and after aerosolization*. Analytical Methods. 6. 7324-7334

41. Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. ArXiv preprint at arxiv:1609.04747.

42. Conniffe, D. (1987). *Expected Maximum Log Likelihood Estimation*. The Statistician. 36. 317-329

43. Aldrich, J. (1997). *R.A. Fisher and the making of maximum likelihood 1912-1922*. Statistical Science. 12. 162-176

44. Aurelio, Y.S., de Almeida, G.M., de Castro, C.L., und Braga, A.P. (2019). *Learning from Imbalanced Data Sets with Weighted Cross-Entropy Function*. Neural Processing Letters. 50. 1937-1949

45. Kline, D.M. und Berardi, V.L. (2005). *Revisiting squared-error and cross-entropy functions for training neural network classifiers*. Neural Computing & Applications. 14. 310-318

46. Kingma, D.P. und Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. ArXiv preprint at arxiv:1412.6980.

47. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., und Bengio, Y. (2014). *Generative adversarial nets*. Advances in neural information processing systems. 27.

48. Zhu, J.-Y., Park, T., Isola, P., und Efros, A.A. (2017). *Unpaired image-to-image translation using cycle-consistent adversarial networks*. In Proceedings of the IEEE CVPR 2017. 2223-2232

49. He, K., Zhang, X., Ren, S., und Sun, J. (2015). *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. In Proceedings of the IEEE CVPR 2015. 1026-1034

50. Glorot, X. und Bengio, Y. (2010). *Understanding the difficulty of training deep feedforward neural networks*. In Proceedings of the AISTATS. 249-256

51. Mahendran, A. und Vedaldi, A. (2015). *Understanding deep image representations by inverting them*. Auflage. 5188-5196

52. LeCun, Y., Bottou, L., Bengio, Y., und Haffner, P. (1998). *Gradient-based learning applied to document recognition*. IEEE CVPR 1998. 86. 2278-2324

53. Nair, V. und Hinton, G. (2010). *Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair*. Auflage. 807-814

54. Huang, G., Liu, Z., Maaten, L.V.D., und Weinberger, K.Q. (2017). *Densely Connected Convolutional Networks*. In Proceedings of the IEEE CVPR 2017. 2261-2269

55. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., und Liang, J. (2018). *Unet++: A nested u-net architecture for medical image segmentation*. In: Deep learning in medical image analysis und multimodal learning for clinical decision support. 1. Auflage. Springer. 3-11

56. Ronneberger, O., Fischer, P., und Brox, T. (2015). *U-net: Convolutional networks for biomedical image segmentation*. In Proceedings of the MICCAI 2015. 234-241

57. Shorten, C. and Khoshgoftaar, T.M. (2019). *A survey on Image Data Augmentation for Deep Learning*. Journal of Big Data. 6. 60

58. Perez, L. und Wang, J. (2017). *The Effectiveness of Data Augmentation in Image Classification using Deep Learning*. ArXiv preprint at arxiv:1712.04621.

59. Mikołajczyk, A. und Grochowski, M. (2018). *Data augmentation for improving deep learning in image classification problem*. In Proceedings of the IIPhDW 2018. 117-122

60. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., und Isard, M. (2016). *TensorFlow: a system for Large-Scale machine learning*. In Proceedings of the 12th USENIX symposium on operating systems design und implementation (OSDI 16). 265-283

61. Mill, L., Wolff, D., Gerrits, N., Philipp, P., Kling, L., Vollnhals, F., Ignatenko, A., Jaremenko, C., Huang, Y., De Castro, O., Audinot, J.-N., Nelissen, I., Wirtz, T., Maier, A., und Christiansen, S. (2021). *Synthetic Image Rendering Solves Annotation Problem in Deep Learning Nanoparticle Segmentation*. Small Methods. 5. 2100223

62. Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., und Liang, J. (2019). *UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation*. IEEE TMI 2019. 39. 1856-1867

63. Bank, D., Koenigstein, N., und Giryes, R. (2020). *Autoencoders*. ArXiv preprint at arxiv:2003.05991.

64. Muñoz-Mármol, M., Crespo, J., Fritts, M.J., und Maojo, V. (2015). *Towards the taxonomic categorization and recognition of nanoparticle shapes*. Nanomedicine. 11. 457-465

65. Ul-Hamid, A. (2018). *A Beginners' Guide to Scanning Electron Microscopy*. 1. Auflage. Springer Nature Switzerland AG

66. TheGIMPDevelopmentTeam. (2019). *GIMP* 2.10.12. https://www.gimp.org. Letzmalig Besucht am 12.04.2023

67. Rühle, B., Krumrey, J.F., und Hodoroaba, V.-D. (2021). *Workflow towards automated segmentation of agglomerated, non-spherical particles from electron microscopy images using artificial neural networks*. Scientific Reports. 11. 4942

68. Jain, V., Bollmann, B., Richardson, M., Berger, D.R., Helmstaedter, M.N., Briggman, K.L., Denk, W., Bowden, J.B., Mendenhall, J.M., Abraham, W.C., Harris, K.M., Kasthuri, N., Hayworth, K.J., Schalek, R., Tapia, J.C., Lichtman, J.W., und Seung, H.S. (2010). *Boundary Learning by Optimization with Topological Constraints*. In Proceedings of the IEEE CVPR 2010. 2488-2495

69. Bals, J., Loza, K., Epple, P., Kircher, T., und Epple, M. (2022). *Automated and manual classification of metallic nanoparticles with respect to size and shape by analysis of scanning electron micrographs*. Materialwissenschaft & Werkstofftechnik. 53. 270-283

70. Reed, S.J.B. (2005). *Electron Microprobe Analysis and Scanning Electron Microscopy in Geology*. 2. Auflage. Cambridge University Press

71. Wall, E., Blaha, L.M., Paul, C.L., Cook, K., und Endert, A. (2018). *Four Perspectives on Human Bias in Visual Analytics*. In: Cognitive Biases in Visualizations. 1. Auflage. Springer. 29-42

72. Kahneman, D. (2012). *Schnelles Denken, Langsames Denken*. 1. Auflage. Siedler Verlag

73. BlenderOnlineComunity. (2022). *Blender* 3.3.0. http://www.blender.org. Letzmalig Besucht am 12.04.2023

74. Fend, C., Moghiseh, A., Redenbach, C., und Schladitz, K. (2021). *Reconstruction of highly porous structures from FIB-SEM using a deep neural network trained on synthetic images*. Journal of Microscopy. 281. 16-27

75. Prill, T. und Schladitz, K. (2013). *Simulation of FIB-SEM Images for Analysis of Porous Microstructures*. Scanning. 35. 189-195

76. Isola, P., Zhu, J.-Y., Zhou, T., und Efros, A. (2017). *Image-to-Image Translation with Conditional Adversarial Networks*. In Proceedings of the IEEE CVPR 2017. 5967-5976

77. Han, C., Hayashi, H., Rundo, L., Araki, R., Shimoda, W., Muramatsu, S., Furukawa, Y., Mauri, G., und Nakayama, H. (2018). *GAN-based synthetic brain MR image generation*. In Proceedings of the IEEE ISBI 2018. 734-738

78. Johnson, J., Alahi, A., und Fei-Fei, L. (2016). *Perceptual Losses for Real-Time Style Transfer und Super-Resolution*. In Proceedings of the Computer Vision – ECCV 2016. 694-711

79. Ghafoorian, M., Karssemeijer, N., Heskes, T., van Uden, I.W.M., Sanchez, C.I., Litjens, G., de Leeuw, F.E., van Ginneken, B., Marchiori, E., und Platel, B. (2017). *Location Sensitive Deep Convolutional Neural Networks for Segmentation of White Matter Hyperintensities*. Scientific Reports. 7. 5110

80. Rand, W.M. (1971). *Objective Criteria for the Evaluation of Clustering Methods*. Journal of the American Statistical Association. 66. 846-850

81. Bals, J. und Epple, M. (2023). *Deep learning for automated size and shape analysis of nanoparticles in scanning electron microscopy*. RSC Advances. 13. 2795

82. Dietrich, O., Raya, J., Reeder, S., Reiser, M., und Schoenberg, S. (2007). *Measurement of signal-to-noise ratios in MR images: Influence of multichannel coils, parallel imaging, and reconstruction filters*. Journal of Magnetic Resonance Imaging. 26. 375-385

83. Plaat, A. (2022). *Deep Reinforcement Learning*. 1. Auflage. Springer

84. Deng, J., Dong, W., Socher, R., Li, L.J., Kai, L., und Li, F.-F. (2009). *ImageNet: A large-scale hierarchical image database*. In Proceedings of the IEEE CVPR 2009. 248-255

85. van der Maaten, L. und Hinton, G. (2008). *Viualizing data using t-SNE*. Journal of Machine Learning Research. 9. 2579-2605

86. Joana (Accountname). *Photoshopsupply.* https://www.photoshopsupply.com/patterns-textures/free-dust-textures. Letzmalig Besucht am 22.12.2022.

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

88. Ilett, M., Wills, J., Rees, P., Sharma, S., Micklethwaite, S., Brown, A., Brydson, R., und Hondow, N. (2020). *Application of automated electron microscopy imaging and machine learning to characterise and quantify nanoparticle dispersion in aqueous media*. Journal of Microscopy. 279. 177-184

89. Olejnik, M. (2021). *Synthese von Zinkoxid-, Titandioxid- und Silika-Partikeln mit definierter Form, Größe, Kristallinität und ihre zellbiologischen Effekte*. Dissertation - Universität Duisburg-Essen

8 Anhang

8.1 Technische Spezifikationen

Die hier gezeigten neuronalen Netze wurden final mittels Python 3.9 und Tensorflow 2.10 implementiert. Für die Implementierung der Gewichtungskarten wurde sklearn 0.24.2 verwendet. Die Partikeleigenschaften wurden mittels Open Computer Vision 2 4.5.5 berechnet. Für Input /Output Operationen wurden Numpy 1.20.3 und Pandas 1.3.4 verwendet. Weiterhin wurde Matplotlib 3.4.3 für die Darstellung diverser Ergebnisse genutzt. Die Annotationsmasken wurden mittels GIMP 2.10.12 erstellt.

8.2 Abkürzungsverzeichnis

ADAM	Adaptive Momentum Estimation - Optimierungsalgorithmus
AE	Autoencoder
AlexNet	Netzwerk Architektur entwickelt von Alexander Krizhevsky et al.
AvgPool	Average Pooling - Schichttyp eines faltenden neuronalen Netzes
CAE	Convolutional Autoencoder -Typus eines neuronalen Netzes
CNN	Convoltional Neural Network
Conv.	Convolution - Schichttyp eines faltenden neuronalen Netzes
DC-Module	Dense Convolutional Module
FC	Fully Connected - Schichttyp eines faltenden neuronalen Netzes
FN	False Negative
FP	False Positive
GAN	Generative Adversarial Network
GIMP	GNU Image Manipulation Program
GoogLeNet	Netzwerk Architektur entwickelt von Google
HAADF	High Angle Annular Dark Field
HF	Hellfeld
ILSVRC	Large Scale Visual Recognition Challenge
KI	Künstliche Intelligenz

MaxPool	Maximum Pooling - Schichttyp eines faltenden neuronalen Netzes
ML	Machine Learning
px	Pixel
ReLU	Rectified Linear Unit
REM	Rasterelektronenmikroskopie
ResNet	Residual Network
ResNet34	34 Schichten tiefe Residual Network Architektur entwickelt von He et al.
RGB	Rot Grau Blau - Farbkanäle eines Bildes
s	Stride - Schrittweite eines Filters beim Abtasten eines Bildes
SE	Secondary Electron
STEM	Scanning Transmission Electron Microspopy
TN	True Negative
TP	True Positive
UNet	U-förmige Netzwerk Architektur für die Bildsegmentierung
UNet++	U-förmige Netzwerk Architektur für die Bildsegmentierung nach 2-facher Weiterentwicklung
VGG	Synonym für eine Netzwerkarchitektur; Teamname von K. Simonyan & A. Zisserman welches beim ImageNet ILSVRC-2014 Wettbewerb des ersten Platz erreichte

8.3 Publikationsliste

Publikationen in Fachzeitschriften

J. Bals, K. Loza, P. Epple, T. Kircher and M. Epple, (2022), “Automated and manual classification of metallic nanoparticles with respect to size and shape by analysis of scanning electron micrographs”, Materialwissenschaft & Werkstofftechnik, 53, 270-283

J. Bals & M. Epple, (2023), “Deep learning for automated size and shape analysis of nanoparticles in scanning electron microscopy”, RSC Advances, 13, 2795-2802

J. Bals & M. Epple, (2023), “Artificial SEM images created by generative adversarial networks (GAN) from simulated particle assemblies”, Advanced Intelligence Systems, 2300004

Beiträge zu wissenschaftlichen Tagungen

Materials Week 2021 Virtual Congress & Exhibition

07. - 09.09.2021 (Online, Poster)

J. Bals, K. Loza, M. Epple, „When is a triangle a triangle?”

17th GCC - German Conference on Cheminformatics

08. - 12.05.2022 (Garmisch-Partenkirchen, Poster)

J. Bals, M. Epple, „Automated Analysis of SEM Images of Nanoparticles by Deep Learning”

MRS Fall Meeting & Exhibition 2022

28.11. - 02.12.2022 (Boston, MA, USA, Vortrag)

J. Bals & M. Epple, „Synthetic Scanning Electron Micrographs Created by Generative Adversarial Networks (GANs) from Artificial Height Maps“

DGK Jahrestagung

27.03. – 30.03.2023 (Frankfurt am Main, Vortrag)

J. Bals & M. Epple, „The generation of artificial SEM images of nanoparticles by deep neural networks”

9 Lebenslauf

Der Lebenslauf ist aus rechtlichen Gründen nicht einsehbar.

10 Danksagung

Ich danke Herrn Prof. Matthias Epple für die Stellung des Themas, die Bereitschaft einem Fachfremden die Möglichkeit zu geben eine neue berufliche Richtung einzuschlagen und die Geduld bei der Betreuung der Arbeit.

Ich danke meiner Familie, Freunden, und meiner Freundin für die Unterstützung, Geduld, ein offenes Ohr, die Bereitschaft Sorgen zu teilen und vieles mehr.

Ich danke meinen Kolleginnen und Kollegen für eine tolle Arbeitsumgebung, die Bereitschaft füreinander da zu sein und einander zu unterstützen.

11 Eidesstattliche Erklärung

Ich bestätige hiermit, dass ich die vorliegende Arbeit mit dem Titel „Synthese und Analyse elektronenmikroskopischer Aufnahmen mittels künstlicher Intelligenz“ selbstständig und nur mit den angegebenen Hilfsmitteln angefertigt habe. Die Arbeit wurde in dieser oder ähnlicher Form noch bei keiner anderen Universität eingereicht.

Bochum, den 10.02.2023

_____

Jonas Bals