Universität Bremen

Fraunhofer IFAM

Entwicklung und Analyse von Machine-Learning-Modellen zur Identifikation von Flugzeuglackschichten auf Bildern einer Hyperspektralkamera

Masterarbeit
im Fachbereich Informatik
am Fraunhofer IFAM

von

Jannes Adam
jadam@uni-bremen.de

Betreuer:
Dr. René Neuholz
Tim Strohbach

Gutachter:
Prof. Dr.-Ing. Udo Frese
Dr. Hauke Brüning

30. Mai 2022

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Bremen, den 30. Mai 2022

Zusammenfassung

Zum effizienten Entfernen von Lackschichten per Laser ist die Identifikation des genauen Lacks nötig, um den Laser entsprechend parametrisieren zu können. Eine Hyperspektralkamera ist prinzipiell dazu geeignet, Lacke zu bestimmen. Dafür ist jedoch auch ein Modell nötig, das die aufgenommenen Spektren klassifiziert.
Diese Arbeit beschäftigt sich daher mit der Suche nach geeigneten Methoden und Modellen, um Lackschichten unabhängig von ihrer Farbe identifizieren zu können. Auf Basis von Literaturrecherchen werden dazu Vorverarbeitungsmethoden implementiert und Random Forests sowie Neuronale Netze für die Modelle ausgewählt. Die Modelle werden mit einem dafür erstellten Datensatz von zwölf Lacken - darunter Primer, Basecoats, Clearcoats und Topcoats - trainiert. Auf den Proben sind sowohl frisch lackierte Oberflächen vorhanden als auch welche, die per Laser bearbeitet wurden.
Um zu untersuchen, ob die verbleibende Schichtdicke näherungsweise bestimmt werden kann, werden sowohl Modelle trainiert, die nur den Lack identifizieren, die zwischen gelasert und ungelasert unterscheiden als auch die, die Anzahl der Laserzyklen bestimmen sollen.
Auf der Suche nach einem geeigneten Modell werden bei den Random Forests Hyperparameter und bei den neuronalen Netzen die Architektur variiert. Die trainierten Modelle werden in Hinblick auf Korrektheit und Geschwindigkeit bewertet und die Entscheidungen mit Hilfe von Methoden der erklärbaren künstlichen Intelligenz analysiert.
Sowohl mit den Random Forests als auch mit den neuronalen Netzen ist eine gute Erkennung der Lackschichten möglich. Die neuronalen Netze erreichen aber etwas bessere Ergebnisse. Eine wichtige Erkenntnis ist, dass die Differenzierung der Klassen in gelaserte und ungelaserte Oberflächen bzw. sogar die einzelnen Laserzyklen zu einer Verschlechterung der Metriken, jedoch zu einer besseren Erkennung der Basecoats führt.
Die Geschwindigkeit des neuronalen Netzes konnte gegenüber der Vorlage aus der Literatur deutlich erhöht werden, sodass sowohl einige Random Forests als auch ausgewählte neuronale Netze in der Lage sind, die anfallenden Daten mit der nötigen Framerate zwischen $50Hz$ und $200Hz$ zu klassifizieren.
Einblicke in das Modell zu erhalten, ist auch wegen der Unklarheit der Zusammensetzung der Spektren nur bedingt möglich. Trotzdem konnten Methoden identifiziert werden, die Erkenntnisse über die Modelle bringen oder einzelne Entscheidungen analysieren können.

Inhaltsverzeichnis

1 Einleitung

1.1 Hintergrund

Flugzeuge werden in mehreren Schichten mit unterschiedlichen Lacken beschichtet, um sie optimal vor den Umwelteinflüssen wie Temperatur, Niederschlag und Partikeln in der Luft und damit vor Korrosion zu schützen. Die Grundlage bildet dabei der Primer, der auf die gereinigte und geschliffene Oberfläche aufgetragen wird und vor Korrosion schützt. Darauf wird ein farbiger Finish-Lack oder ein farbiger Basislack und ein Klarlack aufgetragen, die für die Optik und glatte Oberflächen sorgen.

Sollen die Lackschichten wieder abgetragen werden, so erfolgt dieses bisher mit Chemikalien und Lösungsmitteln oder durch Schleifen mit damit verbundenen Staubemissionen.
Dadurch ist das (Ent-)Lackieren eines Flugzeugs ein sehr aufwendiger Prozess, bei dem zudem gesundheitsgefährliche Stoffe verwendet werden. Um diesen Prozess zu verbessern und zu verhindern, dass diese Arbeit auch wegen dort niedrigerer Lohnkosten ins Ausland verlagert wird und Deutschland als Produktionsstandort zu sichern, wird am Fraunhofer Institut für Fertigungstechnik und Angewandte Materialforschung (IFAM) an einem Projekt geforscht, wie dieser Prozess durch den Einsatz von innovativen Methoden automatisiert werden kann.

Abbildung 1.1: Durch einen Gitterschnitt über eine Niete wurde das Abplatzen des Lacks herbei geführt, sodass das Metall darunter offen liegt.

Um dieses Ziel zu erreichen, wird unter anderem erforscht, wie mit Hilfe eines Lasers der Lack entfernt werden kann. Dieses soll sowohl für große Flächen geschehen als auch für kleine Fehlstellen, bei denen z.B. der Lack teilweise abgeplatzt ist, wie in Abbildung 1.1 zu sehen.
Damit in diesem Fall darunter liegende Lackschichten nicht beschädigt werden, muss bestimmt werden, wo welcher Lack aufgetragen ist, um in dem Bereich nur den Lack zu entfernen, der teilweise abgeplatzt ist.

Außerdem ist die genaue Identifikation der verwendeten Lacke wichtig, um den Laser passend einzustellen. Wie Versuche in dem Projekt gezeigt haben, benötigt jeder Lack eine andere Parametrisierung des Lasers, damit die betreffende Lackschicht fachgerecht entfernt wird. Des Weiteren kann das Scannen der Oberfläche mit einhergehender Identifikation des verwendeten Lacks in der Qualitätssicherung angewendet werden, um Fehlstellen zu identifizieren, die mit dem bloßen Auge nicht zu sehen sind.
Da die Farbe unterschiedlicher Lacke teilweise nicht zu unterscheiden und der Klarlack transparent ist, lassen sich über RGB-Kameras teilweise keine Unterschiede erkennen. In Vorversuchen konnte aber gezeigt werden, dass sich die Lacke mit einer Nahinfrarot (NIR) - Hyperspektralkamera unterscheiden lassen. Am IFAM wurde dazu ein Prüfkopf für einen Roboter konstruiert, der eine Specim FX17e und geeignete Beleuchtung enthält. Es fehlt jedoch noch ein geeignetes Modell, das die Lackschichten identifizieren kann und alle Anforderungen erfüllt. Die bisher verwendeten Modelle, die mit Hilfe einer Software zur Verarbeitung hyperspektraler Daten erstellt wurden, kommen bereits bei sehr wenigen Lacken an ihre Grenzen und ihre Klassifikation ist nur bedingt verlässlich.

1.2 Zielsetzung

Das Ziel dieser Masterarbeit ist die Implementierung eines Systems, das in der Lage ist, die oberste Lackschicht (bzw. bei Klarlacken zusätzlich den darunter liegenden Basecoat) zu identifizieren, um diese Information für die Laserentlackung bereit zu stellen.
Dazu wurden einige Lacke ausgewählt (siehe Kapitel 3.2), bei denen teilweise erwartet wird, dass sie schwer zu differenzieren sein werden. Dazu gehören Lacke gleicher Farbe, die mit dem Auge nicht zu unterscheiden sind; sehr dunkle Lacke, bei denen eine geringe Intensität der Reflektion erwartet wird und Klarlacke.
Für die Klassifikation der Daten der Hyperspektralkamera sollen Machine-Learning-Modelle erstellt und hinsichtlich ihrer Eignung verglichen werden. Da Machine-Learning-Modelle datenbasiert trainiert werden, muss zunächst ein gelabelter Datensatz mit allen möglichen Oberflächen in ausreichender Größe erstellt werden. Die Modelle müssen zwar keine Echtzeitanforderungen erfüllen, sollten die Bewertung aber in etwa der Zeit durchführen können, in der die Bilder aufgenommen werden.
Wegen hoher Anforderungen in der Luftfahrt zu der Erklärbarkeit von Algorithmen, soll analysiert werden, wie die Modelle zu ihren Entscheidungen kommen und ob sich daraus zum Beispiel Regeln oder Gemeinsamkeiten ableiten lassen.
Die Entwicklung dieser Modelle, deren Grundlage und deren Analyse wird in der vorliegenden Arbeit dokumentiert. Abschließend wird beurteilt, ob und welche der entwickelten Modelle die Lacke zuverlässig unterscheiden können oder wo ihre Grenzen liegen.

1.3 Aufbau der Arbeit

Kapitel 2 stellt die Eigenschaften hyperspektraler Bildgebung und deren Besonderheiten bei der Vorverarbeitung vor. Zudem wird auf Modelle eingegangen, die sich zur Klassifizierung der Hyperspektralbilder eignen.
In Kapitel 3 werden Hintergrund und Ziele des dieser Masterarbeit zu Grunde liegenden Forschungsprojekt am Fraunhofer IFAM vorgestellt. Außerdem werden die

vorhandenen Proben und Lacke sowie die verwendete Hardware beschrieben und im Kontext der Projektziele und der Erkenntnisse aus Kapitel 2 Entscheidungen für die zu untersuchenden Machine-Learning-Algorithmen getroffen.
Diese Algorithmen und die verwendeten Metriken zur Auswertung der Korrektheit und Erklärbarkeit werden in Kapitel 4 vorgestellt.
Kapitel 5 beschäftigt sich mit der konkreten Umsetzung der ausgewählten Machine-Learning-Modelle und beschreibt unter anderem, welche Frameworks und Architekturen ausgewählt wurden.
Die trainierten Modelle werden in Kapitel 6 hinsichtlich Korrektheit, Geschwindigkeit und Erklärbarkeit analysiert. Ein Vergleich und eine Bewertung in Hinsicht auf das Projektziel findet in Kapitel 7 statt.
Die erlangten Kenntnisse und sich daraus ergebenden Fragen werden in Kapitel 8 zusammengefasst.
Die für einen Absatz verwendeten Quellen werden an dessen Ende angegeben. Die Arbeit ist im passiven Schreibstil formuliert. Sofern nicht explizit angegeben, wurden aber alle beschriebenen Arbeiten und Erkenntnisse im Rahmen dieser Arbeit getätigt und erlangt.

2 Verarbeitung hyperspektraler Bilddaten

2.1 Hyperspektrale Bildgebung

Hyperspektralbilder sind Abbildungen mit kontinuierlichem Spektrum. Im Gegensatz dazu versuchen Farbbilder, das menschliche Sehen mit meist drei spektralen Kanälen nachzubilden. Aus diesen drei Kanälen können die verschiedenen Farben nachgebildet werden.
Vorgänger und Spezialfall von Hyperspektralbildern sind multispektrale Aufnahmen. Dabei werden mehrere einkanalige Bilder für unterschiedliche Wellenlängen mit Hilfe von Filtern oder entsprechender Beleuchtung aufgenommen. So lässt sich daraus ein Bild mit mehreren spektralen Kanälen, die aber meist nicht kontinuierlich sind, aufnehmen.
Hyperspektralaufnahmen können für unterschiedliche Wellenlängenbereiche angefertigt werden. Da für jedes Pixel ein kontinuierliches Spektrum gemessen wird, erhalten die Daten neben der räumlichen noch eine spektrale Dimension. Man spricht daher bei den aufgenommenen Daten von einem Hypercube mit den Dimensionen $X \times Y \times \lambda$ [1].
Hyperspektralkameras sind jedoch oft Zeilenkameras. Das heißt, sie nehmen kein Bild mit zwei räumlichen Dimensionen auf, sondern eines mit je einer räumlichen und einer spektralen Dimension. Damit trotzdem räumlich-zweidimensionale Bilder entstehen, können mehrere Zeilen, die nacheinander aufgenommen wurden, während die Kamera oder das aufgenommene Objekt bewegt wurde, aneinandergehängt werden. Hyperspektralkameras messen die diffuse Reflektion von Oberflächen. Die Beleuchtung dazu kann durch das Umgebungslicht - also durch die Sonne oder das Raumlicht - oder über eine zielgerichtete Beleuchtung mit Halogen- oder Xenon-Lampen oder LEDs geschehen [2].
Der Vorteil von hyperspektraler Bildgebung gegenüber gewöhnlichen Farbbildern ist, dass Materialien - ohne deren Farbe zu nutzen - voneinander unterschieden werden können. Dabei wird das Prinzip der Spektroskopie verwendet, bei der Materialien durch ihr charakteristisches Spektrum nach Bestrahlung mit elektromagnetischen Wellen bestimmt werden können. Analog dazu ist das reflektierte Spektrum eines

Materials im NIR-Bereich charakteristisch für seine Zusammensetzung. So können Materialien durch den Vergleich mit Referenzspektren identifiziert werden. Zwar ist die Genauigkeit der hyperspektralen Bildgebung - beispielsweise durch Rauschen und Interferenzen benachbarter Materialien - nicht so hoch wie bei einer Spektroskopie im Infrarot-Bereich, die Geschwindigkeit, mit der große Flächen aufgenommen werden können, ist dafür aber deutlich höher [1, 3, 4].

2.2 Vorverarbeitung von Hypercubes

Verschiedene Umstände machen die Vorverarbeitung von hyperspektralen Bildern wie auch bei anderen Daten sinnvoll. Hyperspektralkameras messen die Spannung der einzelnen Pixel, die durch das einfallende Licht entstehen. Diese Spannung ist abhängig von der Lichtintensität und der Beleuchtungszeit und daher für sich stehend wenig aussagekräftig. Daher ist der übliche Weg eine Umrechnung in den Reflexionsgrad, der in % angegeben wird. Diese Umrechnung geschieht mit einer Kalibrierung. Für deren Berechnung werden normalerweise Reflexionsstandards verwendet, die zu bestimmten Prozentzahlen das einfallende Licht gleichmäßig reflektieren. Dafür wird meist gepresstes Polytetafluoroethylene-Puder - auch Spectralon genannt - genutzt. Die einfachste Kalibrierung ist die Onepoint-Kalibrierung, bei der angenommen wird, dass sich der Sensor linear verhält. Für die Onepoint-Kalibrierung werden zunächst zwei Bilder aufgenommen: das Schwarzbild B und die Weißreferenz W. Das Schwarzbild gibt an, wie viel Spannung der Sensor generiert, wenn kein Licht einfällt und wird üblicherweise bei geschlossenem Kamerashutter mit der ausgewählten Aufnahmezeit aufgenommen. Für die Weißreferenz wird ein Bild des Kalibrierungsstandards, welcher 99% des Lichts reflektiert, unter der gewählten Beleuchtung und mit der definierten Aufnahmezeit aufgenommen. Der Reflexionsgrad X in % wird für eine Aufnahme S nach 2.1 berechnet.

$$X = 100(S - B)(W - B)^{-1} \tag{2.1}$$

Da Spectralon mit $2,2\frac{€}{cm^2}$ ziemlich teuer ist, werden alternative Kalibrierungsmöglichkeiten gesucht. Sollen nur spektrale Signaturen wiedererkannt werden, ist die Skalierung des Reflexionsgrads irrelevant. Daher kann dafür auch ein anderes Material als Kalibrierungsstandard verwendet werden. Es ist jedoch wichtig, ein Material zu wählen, dessen Oberfläche die relevanten Wellenlängen möglichst gleichmäßig und diffus reflektiert. Außerdem sollte es keine Feuchtigkeit aufnehmen, da das die Reflektion beeinflussen würde [5, 2].
Wie auch bei anderen Sensoraufnahmen kann Rauschen die Datenaufnahme beeinträchtigen. Das Rauschen kann dabei durch vielfältige Quellen entstehen: Eine Möglichkeit ist die Veränderung des einfallenden Lichts durch die Sonne oder Schwankungen bei der Lichtquelle. Auch die Kamera selbst kann Rauschen erzeugen.
Um für Verbesserung zu sorgen, eignen sich die üblichen Methoden zur Rauschunterdrückung wie der Median-Filter oder auch der in der Spektroskopie verbreitete Savitzky-Golay-Filter. Dieser führt eine stückweise Polynomregression auf den Daten durch.
Weitere Anwendungsmöglichkeiten für diese Filter sind fehlende Daten durch tote Pixel oder Probleme beim Aufnehmen einzelner Zeilen [6].
Ein zusätzliches Problem ist die Verzerrung der aufgenommenen Daten durch die

Linse, die Oberflächenkrümmung oder auch nur durch unterschiedliche Abstände der Linse zum Mittelpunkt und zu dem Rand des aufgenommenen Bereichs. Sowohl für diese Probleme - auch Keystone- und Smile-Effekt genannt - als auch für fehlerhafte Pixel besitzen einige Kameras Algorithmen, die die aufgenommenen Daten direkt korrigieren können [7].
Weitere Parameter, die die Aufnahme von Hyperspektralbildern durch Veränderungen in den Reflexionen beeinflussen können, sind die physikalischen Eigenschaften der Oberfläche und insbesondere die Rauheit. Um die Auswirkung dieser Streuung zu reduzieren, können Ableitungen gebildet werden [6].
Ein weiteres Problem, dass bei hyperspektralen Daten auftreten kann, ist dass die Daten durch die hohe spektrale Auflösung zu komplex für (schnelle) Klassifizierungen sind. Eine Reduzierung der Features kann die Dauer des Trainings verkürzen, Overfitting vermeiden, die Genauigkeit der Bewertung steigern und für weniger komplexe Modelle sorgen. Die Feature-Auswahl kann mit verschiedenen Methoden erfolgen, die in drei Kategorien unterteilt werden können: Filter, Wrapper und eingebettete Methoden.
Filter sind dabei unabhängig vom Klassifikator und werden vor dem Training angewendet. Für jedes Feature wird ein Score berechnet. Dieser betrachtet zum Beispiel wie stark ein Feature mit den anderen korreliert und kann so bei der Auswahl von aussagekräftigen Features helfen.
Bei Wrappern wird der Klassifikator als Blackbox betrachtet. Mit Hilfe von Suchalgorithmen werden mögliche, vielversprechende Kombinationen von Features ausgewählt und evaluiert. Da die Suche NP-hart ist und das Modell für jede Evaluation trainiert werden muss, sind diese Methoden jedoch sehr rechenintensiv.
Bei den eingebetteten Methoden ist die Feature-Auswahl ein Teil des Trainingsalgorithmus [8].

2.3 Klassifizierung hyperspektraler Daten

Wie auch bei anderen Problemen, die mit Machine Learning gelöst werden sollen, kommt im Prinzip fast jeder Algorithmus des maschinellen Lernens in Frage. Einige Algorithmen eignen sich aufgrund ihrer Eigenschaften jedoch besser für hyperspektrale Daten und tauchen daher in vielen Veröffentlichungen auf. An dieser Stelle werden die wichtigsten Erkenntnisse einiger Arbeiten vorgestellt, die mehrere Algorithmen ausprobiert und verglichen haben.
Die ausgewählten Artikel nutzen zur Evaluierung ihrer Ansätze Datensätze, die Luftaufnahmen mit Hyperspektralkameras unterschiedlicher Landschaften beinhalten.
Lv und Wang geben einen groben Überblick über verschiedene Klassifizierungsmethoden jedoch ohne Details über die Parameter der Modelle zu geben. Sie verwenden einen Datensatz der Pavia Universität mit Luftaufnahmen von städtischem Gebiet. Sie vergleichen zunächst Modelle, die die Daten jeweils nur auf spektraler Ebene auswerten. Zum Vergleich wurde eine Support Vector Machine (SVM), ein Random Forest, ein Fully Connected Neural Network und ein 1D-Convolutional Neural Network (CNN) trainiert. Zwar wird erwähnt, dass SVMs ein Problem mit einer großen Anzahl von Features haben und nicht gut mehr als zwei Klassen bewerten können, jedoch wird nicht erläutert, wie mit diesen Problemen umgegangen wird. Auch werden die Architekturen der neuronalen Netze nicht vorgestellt. Bei ihren Modellen schneidet das Fully Connected Neural Network mit einer Genauigkeit der Vorhersage mit 80.5% am besten ab. Die SVM und das 1D-CNN liegen mit 80.5%

knapp dahinter. Bei der Genauigkeit schneidet der Random Forest mit 71.5% am schlechtesten ab.
Außerdem vergleichen die Autoren, ob es einen Unterschied macht, ob die Modelle nur die spektralen oder nur die räumlichen Informationen oder beide zusammen verwerten können. Dazu nutzen sie neuronale Netze, da diese in der Lage sind spektrale und räumliche Informationen parallel einzubeziehen. Für diesen Vergleich nutzen die Autoren ein CNN, ein Deep Belief Network (DBN) und ein Netz, das als Stacked Autoencoder (SAE) vortrainiert wird. Während bei den Modellen, die die Daten spektral auswerten nur die eindimensionalen Vektoren betrachtet werden, werden die Daten für die räumliche Auswertung zunächst auf den zwei räumlichen Dimensionen per 2D-Faltung komprimiert, um weitere räumliche Informationen zu extrahieren. Die gleichzeitige spektrale und räumliche Verarbeitung erfolgt über eine 3D-Faltung, die die Informationen aller drei Dimensionen gleichzeitig verrechnet.
Das beste Ergebnis bei diesem Experiment wird durch das CNN bei paralleler spektraler und räumlicher Auswertung mit einer Genauigkeit von 99.8% erreicht. Werden nur die spektrale oder die räumliche Dimension von dem Modell betrachtet, so konnte das per SAE vortrainierte Modell mit 96.5% bzw. 98.7% die besten Ergebnisse erzielen. [9].
Eine detaillierte Beschreibung wie Daten für Modelle vorbereitet und die Modelle parametrisiert werden können, geben Archibald und Fann für SVMs und Hu et al für CNNs.
SVMs haben den Vorteil, dass sie einfach und robust sind. Allerdings können sie nur binäre Entscheidungen treffen. Um trotzdem Probleme mit mehreren Klassen zu lösen, können mehrere SVMs parallel trainiert werden. Diese müssen dann jeweils nur die binäre Entscheidung treffen, ob es sich bei den Daten um eine ausgewählte Klasse handelt oder zwischen zwei ausgewählten Klassen entscheiden. Archibald und Fann verwenden dabei die Strategie, mehrere SVMs für die Entscheidung *eine gegen alle anderen* zu trainieren. Desweiteren besteht bei SVMs das Problem, dass die Berechnungskosten mit der Anzahl der Features quadratisch steigen. Da die spektrale Auflösung von Hyperspektralaufnahmen bei modernen Sensoren hoch ist, sollte die Anzahl der Features reduziert werden.
Die Autoren vergleichen dafür die Recursive Feature Elimination (RFE) mit einer eigens entwickelten eingebetteten Methode. Bei der RFE werden so lange Features entfernt, bis die vorgegebene Anzahl verbleibt. Dazu minimiert die RFE bei SVMs den Generalisierungsfehler.
Ihre eigene, eingebettete Methode errechnet die Gewichtung der Features durch ein logistisches Maß der Wichtigkeit und verwendet diese Gewichtung im Kernel der SVM. Dadurch werden die Features nicht direkt entfernt und die Gewichte können weiter angepasst werden. Erreicht das Gewicht einen Wert unterhalb eines Schwellwertes, so wird es aussortiert und die SVM neu trainiert. Damit kann im Vergleich zur RFE die Genauigkeit gesteigert und Berechnungszeit gespart werden [10].
Welche Architektur für ein CNN für die Verarbeitung hyperspektraler Daten geeignet ist, untersuchen Hu et al. Sie entwickeln dazu eine eigene Architektur und vergleichen deren Genauigkeit und Berechnungszeit mit einer SVM und mehreren anderen Architekturen neuronaler Netze. Dazu verwenden sie die Daten aus drei Datensätzen von Luftaufnahmen.
Ihre Architektur besteht aus einer Faltungsschicht, gefolgt von einem Max-Pooling und einer vollständig verbundenen Schicht. Die Anzahl der Kanäle der Faltungsschicht ist festgelegt auf 20 und die vollständig verbundene Schicht hat 100 Neuronen.

Die Breite des Kernels wird abhängig von der Kanalanzahl der Eingabedaten n_1 gewählt und ergibt sich durch $k_1 = \lfloor \frac{n_1}{9} \rfloor$. Die Breite des Pooling-Kernels wird von den Autoren durch $k_2 = \lfloor \frac{n_1 - k_1 + 1}{n_3} \rfloor$ festgelegt, wobei für n_3 ein Wert zwischen 30 und 40 gewählt wird.
Die so bestimmten Architekturen konnten mit allen drei Datensätzen mit Genauigkeiten zwischen 90.1% und 92.6% bessere Ergebnisse als die SVMs erreichen, deren Werte zwischen 87.6% und 91.7% lagen. Auch im Vergleich mit anderen neuronalen Netzen wie dem LeNet-5 kann die vorgeschlagene Architektur eine höhere Genauigkeit erzielen. Darüber hinaus ist die Berechnung der Inferenz im Vergleich mit den anderen Architekturen in zwei von drei Fällen schneller. Außerdem kommt das Netz im Vergleich zu anderen tiefen neuronalen Netzen mit wenigen Daten aus. Als weitere Verbesserungen schlagen die Autoren z.B. Regularisierung durch Dropout oder eine zusätzliche Betrachtung der räumlichen Dimension vor [11].

3 Anwendungsfallbezogene Modellauswahl

3.1 Projektkontext

Der Lackierungsprozess eines Flugzeugs bei Neubau oder Wartung wird bisher manuell durchgeführt. Vor dem Lackieren ist eine Reinigung der Oberfläche nötig. Dazu wird bisher eine Lösemittelreinigung durchgeführt. Soll außerdem eine Lackschicht entfernt werden, wird diese mit Schleifscheibe und - papier entfernt. Um anschließend den Schleifstaub und andere Rückstände zu entfernen, wird danach eine erneute Lösemittelreinigung durchgeführt. Dieses Vorgehen ist nicht nur sehr aufwendig, es bringt zudem einige Probleme für Mensch und Umwelt mit sich: Emissionen durch Lösemittel und den Schleifstaub gefährden das Personal und die Umwelt. Zudem muss das Verbrauchsmaterial wie die abgenutzten Schleifscheiben entsorgt werden. Zudem ist die Qualität der Oberfläche durch sich abnutzende Schleifscheiben und unkontrollierten manuellen Abtrag nicht konstant.
Einige dieser Probleme können gelöst werden, indem Roboter das Schleifen übernehmen. Diese können jedoch nur einfach gekrümmte Bereiche bearbeiten und müssen die Bereiche um Fenster u.ä. auslassen, um diese nicht zu beschädigen.
Der Lackierungsprozess ist vor allem durch die Maskierung und Demaskierung von Flächen wie Fenstern, die nicht lackiert werden dürfen, sehr aufwendig.
Um allen diesen Problemen und zudem steigenden Vorgaben an Arbeitsplatzsicherheit und Emissionsgrenzwerte und Outsourcing der manuellen Tätigkeiten ins Ausland durch hohen Kosten- und Zeitdruck zu begegnen, forscht das Fraunhofer IFAM zusammen mit mehreren klein- und mittelständischen Unternehmen an innovativen, automatisierten Verfahren und Produktionsschritten. Dafür sollen Endeffektoren für Vorbehandlung, Reinigung, kantenscharfes Lackieren und Qualitätssicherung entwickelt werden. Diese sollen in einer Demonstrationsanlage vereinigt werden, die einen Flugzeugrumpf in Originalgröße bearbeiten können soll.
Das kantenscharfe Lackieren soll entwickelt werden, um auf das Maskieren von Flächen verzichten zu können. Weiterhin soll mit der Anlage eine ortsselektive, bedarfsgerechte Behandlung ermöglicht werden, sodass zum Beispiel kleine Bereiche repariert werden können, anstatt das Flugzeug großflächig neu zu lackieren.

Für viele dieser Schritte ist es nötig, den Zustand der Oberfläche zunächst festzustellen, damit im Anschluss die Behandlung bedarfsgerecht ausgeführt werden kann. Nur so können zum Beispiel Schäden im Lack erkannt werden. Das Entlackieren soll mit einem Laser durchgeführt werden. Bei diesem müssen die Parameter je nach zu entfernendem Lack gewählt werden. Um auch Flächen mit dem Laser zu behandeln, bei denen der Lack teilweise abgetragen ist, muss möglichst genau bestimmt werden, wo sich welcher Lack befindet.
Der verwendete Laser hat einen Durchmesser von $720\mu m$. Da er jedoch mit einer Überlappungsrate von 75% verwendet wird, kann er den Lack mit einer Genauigkeit von $200\mu m$ entfernen. Die Abtastung der Bilder sollte jedoch etwas höher sein, weswegen das aufgenommene Bild mit möglichst hoher Auflösung klassifiziert werden muss. Dabei werden Wahrscheinlichkeiten statt diskreter Vorhersagen bevorzugt, um zu ermöglichen, dass unsichere Bereiche manuell überprüft werden können.
Da der Lack beim Lasern nicht sofort komplett abgetragen wird, muss der Lack auch dann noch erkannt werden, wenn er bereits mit dem Laser bearbeitet wurde. Im Optimalfall kann sogar unterschieden werden zwischen bearbeitetem und unbearbeitetem Lack und so noch genauere Information für die Laserentlackung bereitgestellt werden.

3.2 Lacke und Proben

Durch das Lastenheft des übergeordneten Projekts wurden Luftfahrtlacke definiert, an der sich die Lackauswahl für diese Arbeit orientiert. Dazu gehören Lacke für Zwei- und Dreischichtlacksysteme. Beim Zweischichtlacksystem wird der farbige Topcoat direkt auf den Primer aufgetragen. Bei Dreischichtlacksystemen wird zunächst ebenfalls ein Primer lackiert. Darauf folgt ein farbiger Basecoat und ein schützender Clearcoat.
Die Lackauswahl für dieses Projekt enthält einen weißen Primer, sieben Basecoats in schwarz, grau, gelb und vier Blautönen, zwei Clearcoats und zwei Topcoats in grau und schwarz.
Mit diesen Lacken wurden am Fraunhofer IFAM Proben auf etwa $8*13cm$ großen Prüfblechen angefertigt. Der Lackaufbau erfolgt auch auf den Prüfblechen in Schichten. Einige Proben wurden jedoch gedrittelt, sodass die gesamte Probe mit einem Primer, zwei Drittel davon zusätzlich mit einem Basecoat und davon die Hälfte mit einem Clearcoat lackiert wurden. Andere Probenbleche wurden komplett lackiert. Einige dieser Bleche wurden zudem mit dem Laser in ein bis vier Zyklen bearbeitet, um den Lack abzutragen. Das Entfernen des Lacks hat jedoch sehr unterschiedliche Effekte auf die Proben: Während bei einigen Proben nur sehr wenig Effekt zu sehen ist, wurden andere Lackaufbauten bis auf das Aluminium oder den Primer entfernt. Insbesondere bei den Klarlacken sind außerdem strukturelle Veränderungen zu sehen: Teilweise bildet der Lack Blasen oder er hängt in Fetzen an der Probe. Diese Effekte sind für ausgewählte Proben in Abbildung 3.1 zu sehen.
Die Proben wurden so angefertigt, dass möglichst jeder Clearcoat auf jedem Basecoat aufgetragen wurde, damit der Klarlack auf allen ausgewählten Basislacken erkannt werden kann. Außerdem kann so untersucht werden, ob der darunterliegende Basecoat Auswirkungen auf die Erkennung des Klarlacks hat. Auch für die Laserbearbeitung wurden, soweit das möglich war, alle Kombinationen verwendet, um auch hier Trainingsdaten bereit stellen zu können.
Eine Übersicht über alle Lacke ist im Anhang zu finden: (1)

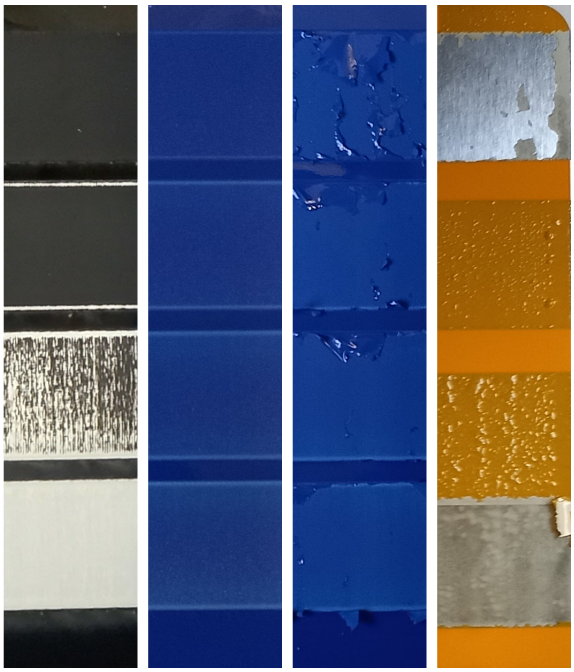


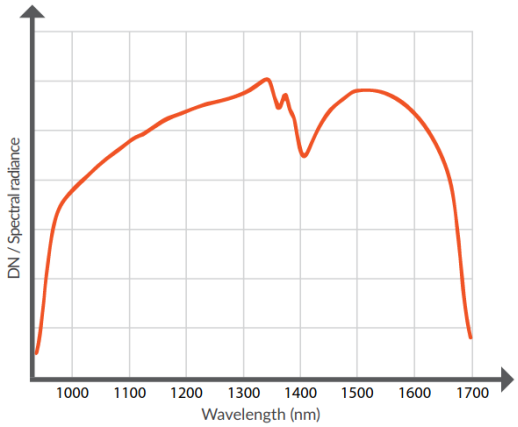
Abbildung 3.1: Abgebildet sind Ausschnitte von vier laserentlackten Proben. Von oben nach unten steigt in den Streifen die Anzahl der Laserzyklen. Alle Proben wurden mit dem Primer grundiert. Bis auf den zweiten Streifen wurden alle Proben mit dem gleichen Klarlack lackiert. Es ist zu erkennen, dass der Effekt des Lasers für jede Lackkombination unterschiedlich ist: Während der Klarlack auf der ersten Probe nach einem Zyklus entfernt und danach der Basecoat abgetragen wurde, hängen bei dem dritten Streifen über alle Zyklen noch einige Fetzen des Klarlacks auf der Probe fest. Beim Streifen ganz rechts wurde der Lack bis auf Reste des Primers bei einem und vier Zyklen komplett abgetragen. Bei zwei und drei Laserentlackungszyklen ist zu erkennen, dass der Klarlack Blasen wirft. Warum hier der Lack bei einem Zyklus bereits nahezu vollständig entfernt wurde, ist nicht bekannt.

3.3 Verwendete Hardware

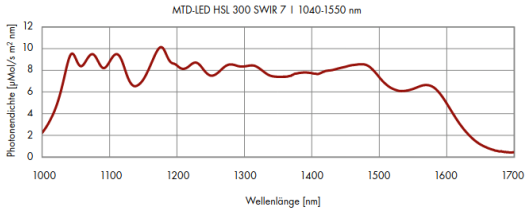
3.3.1 Kamera und Beleuchtung

Alle Aufnahmen dieser Arbeit wurden mit dem am Fraunhofer IFAM konstruierten Prüfkopf aufgenommen. Dieser besteht aus der FX17e [12, 13] - einer Hyperspektralkamera der Firma Specim - und der SWIR-LED-Linienleuchte des Herstellers MTD [14].
Die Specim FX17e ist eine Zeilenkamera, die NIR-Licht im Bereich von $900nm$ bis $1700nm$ in bis zu 224 Bändern aufnehmen kann. Die Breite einer Zeile beträgt $640px$. Die Region of interest (ROI) kann sowohl in räumlicher als auch in spektraler Dimension ausgewählt werden. Wie in Abbildung 3.2a zu sehen ist, ist die aufgenommene Strahlungsdichte nicht konstant über das spektrale Aufnahmeintervall.
Die Bildfrequenz hängt von den eingestellten Parametern ab und kann bis zu 527 Bilder pro Sekunde erreichen. Der Öffnungswinkel der standardmäßig verwendeten Linse beträgt 38°.
Als Schnittstellen zur Kamera stehen CameraLink und GigE Vision - zwei Standard-Schnittstellen aus der industriellen Bildverarbeitung - zur Verfügung. Je nachdem

welche Schnittstelle verwendet wird, kann die FX17e per CameraLink oder Industrie-Ethernet an die verarbeitende Hardware angeschlossen werden. Für die Kommunikation mit der Kamera kann GenICam - eine generische Schnittstelle für Industrie-Kameras - oder ein ASCII-Protokoll verwendet werden.
Die Kamera verfügt über Möglichkeiten zur Onboard-Veränderung der Messungen. Dazu gehören zum Beispiel digitale Verstärker und mehrere Bildkorrekturen wie zum Beispiel automatische Bildverbesserungen und die Ersetzung falscher Pixel.
Da es sich bei der Kamera um eine Zeilenkamera handelt, wurde sie an einem Kuka Roboter angebracht, um Flächen aufnehmen zu können.
Zur Beleuchtung steht eine SWIR-Linienleuchte der Firma MTD zur Verfügung.



(a) Aufgenommene Strahlungsdichte der FX17 in Abhängigkeit der Wellenlänge aus [13]



(b) Emissionsspektrum der Beleuchtung aus [14]

Abbildung 3.2: Aufnahme/Emission der Kamera/Beleuchtung in Abhängigkeit der Wellenlänge

Sie erzeugt SWIR-Licht im Spektralbereich von $1040nm$ bis $1550nm$. Dazu werden zwölf verschiedene Wellenlängen zu einem Spektrum kombiniert. Dadurch ist das Spektrum nicht homogen. Wie in Abbildung 3.2b zu sehen, lassen sich die Wellenlängen einiger LEDs eindeutig erkennen. Um dieses Spektrum anzupassen und für den Verwendungszweck zu optimieren, lassen sich die Wellenlängen getrennt einstellen. Dazu stehen programmierbare Beleuchtungsprofile zur Verfügung.
Der Abstand, der zur Oberfläche eingehalten werden sollte, beträgt 300mm. Damit der beleuchtete Bereich sichtbar gemacht werden kann, besitzt die Beleuchtung eine Pilot-Leuchte, die den mit SWIR-Licht beleuchteten Bereich zusätzlich mit sichtbarem Licht anzeigt.

3.3.2 PC

Für die Versuche wurde ein leistungsfähiger PC verwendet. Dieser besitzt einen Intel Core i9-10980XE, der eine Taktfrequenz von 3GHz und 18 Kerne hat. Der Computer ist ausgestattet mit 128GB Arbeitsspeicher. Insbesondere für die Verwendung der neuronalen Netze besitzt der PC eine NVIDIA Quadro RTX 4000, die über 8GB GPU-Speicher verfügt.

3.4 Vorgehensweise und Auswahl der Modelle

Die Entscheidungen für die Vorverarbeitung der Daten und die Auswahl der Machine-Learning Modelle basiert auf den Anforderungen und den recherchierten Ansätzen für die Verarbeitung hyperspektraler Daten in Kapitel 2.
Die Entwicklung sämtlicher Tools, Modelle und Analysen wurde mit Python getätigt. Insbesondere auch die Aufnahme der Bilder mit der Hyperspektralkamera wird statt durch eine proprietäre Software direkt mit Python gemacht, um eine nicht nachvollziehbare Vorverarbeitung zu vermeiden. Die Smile- und Keystone-Korrektur erfolgt direkt durch die Kamera. Danach werden die Daten kalibriert, um über alle Wellenlängen ähnlich hohe Werte zu erhalten. Für die Kalibrierung wird eine mattweiße Keramikfliese als Low-Cost-Referenz verwendet, da nur das Spektrum identifiziert werden soll.
Für die Modelle gilt, dass ihre Komplexität wegen ähnlicher Spektren und vieler Klassen ausreichend hoch sein muss. Außerdem sollten sie geeignet sein, viele Klassen voneinander schnell zu unterscheiden. Daher wurden zur Untersuchung Random Forests und neuronale Netze ausgewählt. Random Forests schnitten im Vergleich in Kapitel 2.3 zwar am schlechtesten ab, dafür wird aber erwartet, dass sie die Zielgeschwindigkeiten im Gegensatz zu SVMs auch bei vielen Klassen erreichen können.
Random Forests sind komplexe Modelle, die jedoch über eher wenig zu lernende Parameter verfügen und dadurch eher wenige Trainingsdaten benötigen. Neuronale Netze wurden wegen ihrer erfolgreichen Ergebnisse in den recherchierten verwandten Arbeiten und ihrer hohen Geschwindigkeit ausgewählt.
Außerdem haben beide Modellarten den Vorteil, dass die Daten nur wenig vorverarbeitet werden müssen. Dadurch lassen sich die Modelle später etwas leichter erklären, da nur die Modelle und nicht der Zusammenhang mit der Vorverarbeitung nachvollzogen werden muss.
Bei den Random Forests wurden die Anzahl der Bäume und die maximale Anzahl der Blätter als Hyperparameter zum Steuern der Komplexität ausgewählt. Über die Anzahl der Bäume lässt sich die Komplexität erhöhen, und gleichzeitig das Risiko des Overfittings einschränken. Die Mindestanzahl der Blätter pro Baum ergibt sich durch die Anzahl der Klassen. Mit einer höheren Anzahl lässt sich auch hier die Komplexität des Modells von einem Minimum aus erhöhen.
Für die neuronalen Netze wurden mehrere Architekturen ausgewählt, die sich an der Architektur aus Kapitel 2.3 orientieren und in Kapitel 5.3 näher beschrieben sind. Hier ist das Ziel außerdem eine Architektur zu wählen, die Bilder beliebiger Größe verarbeiten kann.
Die entwickelten Modelle werden hinsichtlich ihrer Korrektheit und Geschwindigkeit analysiert. Für ausgewählte Modelle werden außerdem Untersuchungen durchgeführt, die die Entscheidungen der Modelle transparenter machen sollen.

4 Theoretische Grundlagen

4.1 Entscheidungsbäume

Entscheidungsbäume gehören zu dem überwachten Lernen, einem Teilgebiet des maschinellen Lernens. Außerdem lernen sie datenbasiert. Beides zusammen bedeutet, dass sie lernen, indem sie Beispiele unter Angabe der jeweiligen Klasse erhalten, daraus selbstständig Muster extrahieren und zur Vorhersage neuer Datenpunkte nutzen können. Entscheidungsbäume sind sehr flexibel und kommen auch mit mehreren Ausgabeklassen gut zurecht.
Entscheidungsbäume sind zyklenfreie Graphen. Jeder ihrer Knoten ist mit einem Merkmal der Daten und dazugehörigen Bedingungen verknüpft. Bei der Vorhersage eines Datenpunkts - auch Inferenz genannt - kann anhand der Bedingungen entschieden werden, mit welchem Kindknoten fortgefahren wird. Wird dabei ein Blatt - also ein Knoten ohne Kinder - erreicht, so kann an diesem die Vorhersage abgelesen werden. Die Vorhersage basiert auf den Trainingsdaten und entspricht der Klasse, die die Mehrheit der Datenpunkte hatte, die während des Trainings zu eben diesem Blatt zusortiert wurde. Zusätzlich lassen sich die Wahrscheinlichkeiten, mit denen der Datenpunkt zu den jeweiligen Klassen gehört, über die Anteile der Datenpunkte im Blatt berechnen.
Die Suche des optimalen Baums ist ein NP-vollständiges Problem. Daher werden Suchalgorithmen zum Trainieren des Baums verwendet. Die Auswahl des Trainingsalgorithmus bestimmt, ob pro Knoten eine binäre Entscheidung getroffen werden muss oder mehrere Antworten möglich sind.

$$G = 1 - \sum_{k=1}^{n} p_k^2 \tag{4.1}$$

Classification and Regression Trees (CART) ist ein gieriger Algorithmus, der Binärbäume erstellen kann. Das Ziel bei jedem Knoten ist die Aufteilung der Trainingsdaten anhand eines Merkmals und eines Schwellwerts in zwei Gruppen, sodass die Gruppen möglichst rein sind. Dazu wird die Reinheit standardmäßig anhand von Gini berechnet. Für einen Knoten lässt sich der Gini-Wert nach Formel 4.1 berechnen, wobei p_k der Anteil der Datenpunkte von Klasse k an den aufzuteilenden Datenpunkten

im Knoten ist. Die Reinheit nach Aufteilung der Daten auf mehrere Knoten wird berechnet, indem deren Gini-Wert mit dem Anteil der Datenpunkte an der aufzuteilenden Untermenge gewichtet wird.
Nach erfolgreicher Suche nach Merkmal und Schwellwert werden die entstehenden Untermengen rekursiv weiter aufgeteilt bis keine Aufteilung mehr gefunden wird, die die Unreinheit weiter reduzieren kann.
Statt dem Gini-Wert für die Reinheit kann auch die Entropie verwendet werden. Die resultierenden Bäume sind sehr ähnlich, allerdings ist Gini etwas schneller zu berechnen. Die Entropie erzeugt dafür ausbalanciertere Bäume.
Das Problem ist, dass Entscheidungsbäume sehr komplexe Entscheidungen treffen können und damit zu Overfitting neigen und sich die Struktur des Baums zu genau an die Trainingsdaten anpasst. Eine Möglichkeit, dies zu verhindern, ist die Einschränkung der Freiheitsgrade beim Trainieren über Regularisierung. So könnte zum Beispiel die maximale Tiefe des Baums beschränkt werden oder ein Blatt immer eine Mindestanzahl an Datenpunkten beinhalten. Da die Komplexität bei der Vorhersage von der maximalen Tiefe abhängt, kann durch die Beschränkung der Tiefe neben den positiven Effekten gegen das Overfitting die Rechenzeit verkürzt werden. Eine andere Möglichkeit ist das Pruning. Dabei wird zunächst der volle Baum aufgebaut und im Nachhinein geschnitten.
Entscheidungsbäume lassen sich in logische Formeln übersetzen oder visualisieren. Dadurch ist ihr repräsentiertes Wissen sehr leicht extrahier- und interpretierbar und sie gehören zu den White-Box-Modellen [15, 16].

4.2 Random Forest

Random Forest ist ein Klassifikationsalgorithmus, der auf Entscheidungsbäumen basiert und zum Ensemble Learning gehört. Beim Ensemble Learning wird die Vorhersage mehrerer Prädiktoren kombiniert. Durch das Zusammenfassen der Vorhersagen mehrerer Prädiktoren können oft bessere Entscheidungen getroffen werden als durch nur ein Modell. Dies gilt insbesondere dann, wenn die einzelnen Lerner möglichst unabhängig voneinander sind. Um dies zu erreichen, können verschiedene Algorithmen für die Prädiktoren verwendet werden oder es werden die gleichen Prädiktoren mit unterschiedlichen Daten trainiert. Um die Daten aufzuteilen, gibt es zwei Möglichkeiten: Bei Bootstrap-Aggregation - auch Bagging genannt - werden die Teilmengen für das Training der Knoten mit Zurücklegen gebildet und beim Pasting ohne. Bei beiden Methoden können aber trotzdem dieselben Datenpunkte für verschiedene Modelle verwendet werden, bei Bagging jedoch nicht mehrfach für den gleichen Prädiktor. Für das Training eines Random Forest wird Bagging verwendet.
Im Falle eines Random Forests sind die Prädiktoren alle Entscheidungsbäume, deren Vorhersagen zusammengefasst werden. Die Mehrheitsentscheidungen können diskret getroffen werden. Erfolgreicher ist es aber meist, wenn die Wahrscheinlichkeiten der einzelnen Vorhersagen verwendet werden.
Beim Random Forest kann die Diversität der Bäume zusätzlich zum Bagging der Trainingsdaten durch die Beschränkung auf eine zufällige Untermenge der zur Verfügung stehenden Merkmale pro Knoten gesteigert werden.
Die Komplexität der Vorhersage eines Random Forest ist $O(max\_depth \cdot n_{trees})$. Ein Vorteil bei Ensemble-Methoden, das auch auf Random Forests zutrifft, ist jedoch, dass sich die Prädiktoren auf unterschiedlichen CPUs parallel trainieren lassen. Die Durchführung der Inferenz ist ebenso parallel möglich, sodass sich die Rechenzeit

reduzieren lässt.
Die Entscheidungen von Random Forests und deren gelerntes Wissen lässt sich dadurch, dass es sich um mehrere Bäume handelt, nicht so leicht nachvollziehen wie bei einzelnen Entscheidungsbäumen. Die Rechnung, die der Baum durchführt, ist weiterhin leicht überprüfbar, die Entscheidungen aber nicht einfach erklärbar. Um einen ersten Eindruck über das Gelernte zu erhalten, lässt sich aber die relative Wichtigkeit der Merkmale bestimmen. Eine Möglichkeit ist, für jedes Merkmal zu bestimmen, wie stark es die Unreinheit von Knoten im Durchschnitt reduziert. Das Ergebnis wird auch Feature-Importance genannt [15].

4.3 Neuronale Netze

Neuronale Netze sind der Versuch, das Gehirn von Menschen oder Tieren nachzubilden, um so den Computer automatisch komplexe Aufgaben lernen zu lassen. Mögliche Anwendungsgebiete sind die Klassifikation von Daten oder die Segmentierung von Bildern.
Neuronale Netze haben mindestens eine Eingabe- und eine Ausgabeschicht sowie eine Schicht dazwischen. Die Architektur eines Netzes beschreibt ihren Aufbau. Schichten sind oft mit ihren direkten Nachbarschichten verbunden, es sind jedoch auch andere Architekturen möglich, bei denen Schichten zum Beispiel mit mehreren anderen Schichten verbunden sind.
Die Eingabeschicht erhält die meist normalisierten Eingabedaten. Die Ausgabeschicht gibt das Ergebnis der Berechnungen aus. Kategorische Daten - wie zum Beispiel die Klassen bei einer Klassifikation - werden per OneHot codiert, sodass jede Klasse ein eigenes Neuron erhält.
Jede Schicht besteht aus Neuronen wie in Abbildung 4.1 zu sehen. Die Neuronen

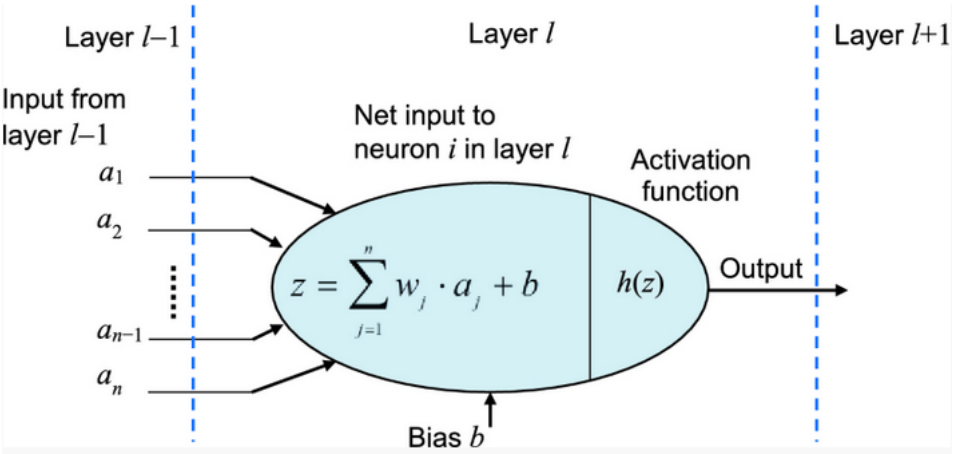


Abbildung 4.1: Einzelnes Neuron mit n Eingängen aus [17]

erhalten die Ausgabewerte der verbundenen Schichten. Dabei kann ein Neuron mit allen Neuronen der verbundenen Schicht verknüpft sein oder nur mit ausgewählten. Das Neuron bildet eine gewichtete Summe über seine Eingänge und addiert einen Bias dazu. Die Gewichte und der Bias sind die zu lernenden Parameter des Netzes. Sie werden zunächst zufällig initialisiert.
Auf die gewichtete Summe jedes einzelnen Neurons wird die Aktivierungsfunktion angewendet. So kann nichtlineares Verhalten erzielt und die Konvergenz verbessert

werden. Es gibt viele mögliche Aktivierungsfunktionen. Dazu zählen zum Beispiel Rectified Linear Unit (ReLU) und Softmax.

$$R(x) = \max(0, z) \tag{4.2}$$

$$a_i(z) = \frac{e^{z_i}}{\sum_{n=1}^{N} e^{z_n}} \tag{4.3}$$

ReLU (siehe Gleichung 4.2) ist dann Null, wenn die gewichtete Summe negativ ist und gibt sonst die gewichtete Summe weiter. Das hat den Vorteil, dass das Training schnell konvergiert, weil viele Neuronen Null ausgeben. Daher wird ReLU viel verwendet.
Softmax (siehe Gleichung 4.3) wird für klassifizierende Ausgabeschichten verwendet. Softmax skaliert die gewichteten Summen der Ausgabeneuronen so auf das Intervall $[0; 1]$, dass die Summe aller Neuronen Eins ergibt und sich die Ausgabewerte als Wahrscheinlichkeiten interpretieren lassen. Die Aktivierungsfunktion der Ausgabeschicht muss je nach Aufgabe des Netzes ausgewählt werden.
Die Rechenschritte in neuronalen Netzen lassen sich als Matrixoperationen ausführen und sind so schnell parallel ausführbar.
Es gibt unterschiedliche Arten von Schichten aus denen neuronale Netze aufgebaut sein können. Dazu gehören vollständig verbundene Schichten, Faltungs- und Pooling-Schichten, die im folgenden vorgestellt werden [17, 18, 19].

4.3.1 Vollständig verbundene Schichten

Bei vollständig verbundenen Schichten sind die Neuronen benachbarter Schichten mit jeweils allen Neuronen der anderen Schicht verbunden. Jedes Neuron ist also mit den i-Eingängen verbunden und hat daher und durch den Bias $i + 1$-Gewichte. Bei n Neuronen ergeben sich dadurch $(i+1)\cdot n$ Parameter, die für die Schicht gelernt werden müssen. Die Anzahl der Neuronen in der Schicht wird durch die Architektur festgelegt. Vollständig verbundene Schichten werden häufig für Klassifikationen eingesetzt [18].

4.3.2 Faltungsschichten

Faltungsschichten werden oft für die Verarbeitung von Bildern verwendet und bilden das menschliche Sehen nach. Dazu wird die Zahl der Neuronen pro Schicht nicht festgelegt, sondern eine Anzahl und Größe von Faltungskernen definiert, die die zu lernenden Gewichte beinhalten. Die Faltungskerne werden mit den Eingabedaten gefaltet. Man spricht statt von Faltungskernen auch von Filtern, da die Faltungskerne durch äquivalente Matrizen ersetzt werden können, die über das Bild geschoben werden und für jeden überdeckten Bereich die Kreuzkorrelation mit dem Filter berechnen (siehe Abbildung 4.2). Die Filter können ein- oder mehrdimensional sein. Durch das Verwenden von Filtern entstehen zwei Vorteile: Zum einen ist die Anzahl der Neuronen nicht festgelegt und zum anderen können Parameter gespart werden. Dadurch, dass die Gewichte einer Schicht nicht den Eingabeneuronen direkt zugeordnet sind, sondern den Filtern, muss die Anzahl der Neuronen pro Schicht nicht von Anfang an feststehen. Das ist besonders für Bilder vorteilhaft, weil so Bilder unterschiedlicher Größe verwendet werden können. Ein Filter errechnet zwar auch die gewichtete Summe, jedoch besitzen sie nicht ein festes Gewicht für die untere linke oder obere rechte Bildecke, sondern weisen zum Beispiel dem linken Pixelnachbar immer wieder dasselbe Gewicht zu. Die Gewichte werden also über alle Pixel *geteilt*, wodurch deutlich weniger zu lernende Parameter entstehen.

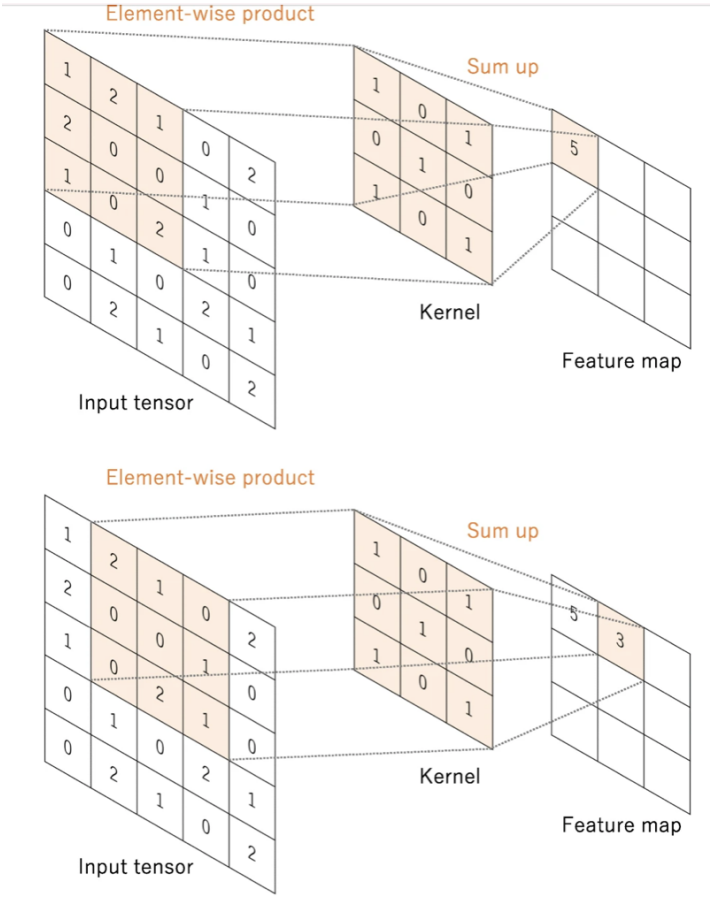


Abbildung 4.2: Ein Beispiel für eine Faltungsoperation, bei dem die Kreuzkorrelation des Eingabebilds mit dem 3*3 Filter jeweils Element für Element berechnet wird. Die Ergebnisse dieses Schrittes werden in der Feature Map gespeichert. Für dieses Beispiel wird kein Padding verwendet und ein Stride von Eins [20].

Für ein oder dreikanalige Bilder wie Fotos werden oft Filter der Größe 3*3 oder 5*5 verwendet. Ein Neuron in der ersten Schicht sieht also nur einen dementsprechend großen Bereich, der auch rezeptives Feld genannt wird. In tieferen Schichten vergrößert sich - vor allem durch Pooling - das rezeptive Feld. So können aus einfachen, lokalen Merkmalen wie Kanten oder Farben in den ersten Schichten komplexe Merkmale in den tieferen Schichten kombiniert werden. Das rezeptive Feld zweier Schichten ist beispielhaft in Abbildung 4.3 visualisiert.
Die Schrittweite - auch Stride genannt - und die Größe des Kernels beeinflussen, wie sehr sich die Ausgabe gegenüber der Eingabe verkleinert. Doch auch bei einer Schrittweite von Eins und einem Faltungskernel von 3*3 würde sich ein Eingabebild verkleinern. Um dies zu verhindern, kann Padding verwendet werden. Dabei wird vor der Faltung das Bild zum Beispiel mit Nullen erweitert, sodass auch für die äußersten Pixelreihen Faltungsergebnisse berechnet werden können.
Das Ergebnis der Faltung eines Kernels mit den Daten wird Feature Map genannt. Normalerweise werden pro Schicht mehrere Filter verwendet, sodass mehrere Feature Maps entstehen [18, 17].

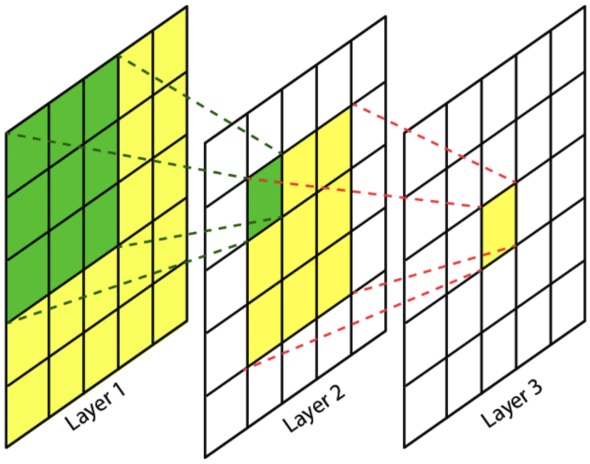


Abbildung 4.3: Es sind zwei Filter mit der Größe 3×3 abgebildet. Während die grüne Fläche das rezeptive Feld eines Pixels des ersten Filters darstellt, zeigt die gelbe Fläche das rezeptive Feld eines Pixels des zweiten Filters über die Schichten hinweg [21].

4.3.3 Pooling

Pooling ist eine Operation, die im Zusammenhang mit Faltung verwendet wird und die Größe von Feature Maps reduziert. Dadurch kann Rechenzeit und Speicherplatz gespart werden. Außerdem kann der Einfluss von Translation und Rotation bei Bildern verringert, sowie Overfitting vermieden werden.
Für Pooling müssen keine Parameter gelernt werden. Es wird ein Kernel mit durch die Architektur definierter Größe und Schrittweite über die Feature Map geschoben. Für jeden ausgewählten Bereich werden die Werte auf eine Zahl reduziert. Ein Bild würde mit einem 2*2-Kernel und einer Schrittweite von Zwei also beispielsweise auf ein Viertel der Größe reduziert.
Beim Pooling stehen mehrere Möglichkeiten zur Verfügung. Am meisten verbreitet ist das Max-Pooling, bei dem jeweils der größte Wert ausgesucht wird. Eine andere Möglichkeit wäre beispielsweise die Mittelwertbildung [17].

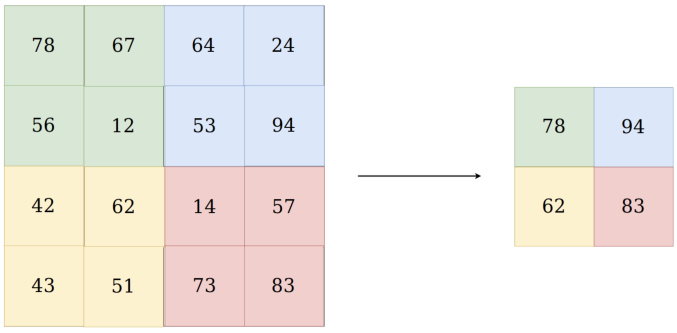


Abbildung 4.4: Max Pooling aus [22]

4.3.4 Autoencoder

Eine spezielle Architekturform für neuronale Netze ist der Autoencoder. Mit diesen Architekturen lassen sich neuronale Netze vortrainieren, ohne dass die Trainingsdaten

dafür annotiert werden müssen. Autoencoder sind neuronale Netze, die die Trainingsdaten nicht nur als Eingabe, sondern auch als Soll-Ausgabe erhalten. Dadurch lernt das Netz eine komprimierte Repräsentation der Eingabedaten, die diese beschreibt und somit die wesentlichen Merkmale beinhaltet. Insbesondere untervollständige Autoencoder, die die Daten in den mittleren Schichten in weniger Dimensionen repräsentieren als die Eingabedaten, erzeugen so eine effiziente Codierung der Merkmale. Mit Hilfe von Transfer Learning kann diese Merkmalsextraktion auf die Schichten eines weiteren neuronalen Netzes übertragen werden, welches auf dieser Basis weitere Aufgaben wie zum Beispiel eine Segmentierung lernen kann. Wenn große Datensätze zur Verfügung stehen, von denen nur kleine Teile annotiert sind, profitieren die Netze so durch höhere Robustheit und bessere Generalisierung [15, 23].
Ein vortrainierter Autoencoder wird verwendet, indem Schichten vom Ende abgeschnitten und durch neue ersetzt werden. Wie Studien gezeigt haben, muss der Schnittpunkt sinnvoll gewählt werden: Je tiefer die Schichten sind, desto spezifischer sind sie an den Datensatz und die Aufgabe angepasst. Allerdings kann die Trennung von Schichten in der Mitte der Netze zu schlechteren Ergebnissen führen, da diese besonders stark aufeinander angepasst sind.
Positiv auf die Performance wirkt sich hingegen die Verwendung ähnlicher Datensätze zum Vortraining aus [24].

4.3.5 Training

Damit ein neuronales Netz für eine Aufgabe geeignete Vorhersagen treffen kann, muss es trainiert werden. Das bedeutet, dass die anfangs zufällig initialisierten Parameter passende Werte erhalten müssen. Dazu werden die Trainingsdaten durchgerechnet und die Ausgabe mit der Referenz verglichen. Die Lossfunktion bewertet die Unterschiede zwischen Soll- und Ist-Ausgabe. Die Lossfunktion - auch Fehlerfunktion genannt - muss passend zur Aufgabe ausgesucht werden. Für die semantische Segmentierung - also die pixelweise Klassifikation - wird oft Cross Entropy, aber auch der Dice-Loss (siehe Kapitel 4.4.6) verwendet [25].
Der errechnete Fehler kann durch Gewichtsanpassungen, welche per Backpropagation errechnet werden, minimiert werden. Dabei wird mit dem Gradientenabstiegsverfahren von der Ausgabeschicht aus rückwärts der Anteil der Neuronen am Fehler berechnet und die Gewichte entgegen des Fehlers angepasst [26].
Die Lernrate legt die Größe der Schritte fest, mit der die Parameter des Modells geändert wird. Sie wirkt sich auf die Dauer und Stabilität des Trainings aus und sollte eines möglicherweise langen Trainings weder zu klein gewählt werden, noch zu groß, da dann die Gefahr besteht, dass das Optimum übersprungen wird. Die Lernrate kann fest gewählt oder durch den Optimierer während des Trainings angepasst werden.
Damit das Training stabiler wird, werden die Daten in Batches zusammengefasst. Die Batchgröße hat ebenfalls Auswirkungen auf die Trainingsdauer und die Stabilität des Trainings. Die maximale Größe der Batches wird durch die Rechenleistung festgelegt. Bei kleiner Größe kann es zu Schwankungen im Lernprozess kommen. In diesem Fall sollte eine kleinere Lernrate gewählt werden, um die Stabilität zu erhöhen. Der vollständige Durchlauf aller Daten bzw. aller Batches wird als Epoche bezeichnet [27, 28].

4.3.6 Regularisierung

Bei neuronalen Netzen kann es wie bei anderen Machine-Learning-Modellen zu Overfitting kommen: Das Netz lernt die Daten auswendig, sodass das Loss beim

Training zwar abnimmt, der Fehler auf den Testdaten aber steigt. Zwar wäre eine Lösung, das Modell weniger komplex zu gestalten, jedoch ist es schwierig, eine optimale Größe zu finden, damit das Modell noch ausreichend komplex ist. Es gibt jedoch mehrere Regularisierungsmethoden, die Overfitting entgegen wirken können. Dazu gehören Dropout und Batch-Normalisierung.
Bei Dropout wird pro Schicht ein Wert gesetzt, der angibt ein wie großer Anteil der Neuronen während des Trainings deaktiviert ist. Dadurch sind zwar mehr Iterationen nötig, dafür wird das Modell aber robuster und die Trainingsdauer pro Epoche sinkt. Meist werden zwischen 20% und 50% der Neuronen deaktiviert [27, 29].
Batch-Normalisierung wird in Form von Schichten angewandt, die in das neuronale Netz oft hinter Faltungsschichten eingefügt werden. Sie stabilisieren das Lernen und beschleunigen es besonders bei tiefen neuronalen Netzen, da höhere Lernraten verwendet werden können. Die Eingabedaten der Batch-Normalisierungs-Schichten werden normiert, sodass für jeden Batch der Mittelwert bei Null und die Varianz bei Eins liegt. Danach werden beide Werte neu eingestellt, indem die Daten skaliert und mit einem Offset versehen werden. Die Parameter dafür werden gelernt [17].
Die Batch-Normalisierung verbessert das Training, da sie die Veränderung der Parameter der vorherigen Schicht kompensiert, die durch die Gewichtsanpassung der Parameter während des Trainings verursacht wird und das Training erschwert [30].

4.4 Qualitätsmaße

Für den Vergleich der Korrektheit von Modellen sind Maße nötig, mit Hilfe derer sie verglichen werden. Dafür wird im ersten Schritt eine Inferenz auf den Testdaten ausgeführt. Das heißt, die Modelle errechnen jeweils ihre Vorhersage für den Testdatensatz. Danach wird das Modell bewertet, indem mit Hilfe der Evaluationsmaße die Vorhersagen mit den Annotationen verglichen werden. Mit diesen Werten ist es möglich, mehrere Modelle zu vergleichen.
Ein Beispiel macht deutlich, dass die Genauigkeit nicht ausreicht, um die Korrektheit von Modellen umfassend zu beschreiben und, dass der Datensatz ebenfalls einen großen Einfluss auf die Evaluation hat: Besteht ein Datensatz aus 100 Datenpunkten, von denen 99 zu Klasse A gehören und nur einer zu Klasse B, würde ein Modell, das immer Klasse A wählt, 99% der Punkte auf dem ausgewählten Datensatz richtig vorhersagen. Bei einem Datensatz mit 50 Beispielen aus Klasse A und 50 aus Klasse B, läge die Genauigkeit hingegen nur noch 50%.
Daher werden im Folgenden mehrere Evaluationsmaße vorgestellt, anhand derer die Korrektheit der Modelle in unterschiedlichen Aspekten bewertet werden kann, sodass auch Datensätze mit einer Ungleichverteilung der Klassen besser beurteilt werden können.

4.4.1 Grundbegriffe

Für die Berechnung einiger Maße müssen die Daten zunächst in vier Klassen eingeteilt werden: True Positive (TP), False Positive (FP), False Negative (FN) und True Negative (TN). Diese Klassifikation funktioniert nur dann direkt, wenn die Vorhersage Wahrheitswerte ausgibt. Gibt das Modell hingegen mehrere Klassen aus, muss eine Klasse ausgewählt oder eine Hypothese aufgestellt werden, nach der die Ausgaben in wahr und falsch klassifiziert werden können.

		Wahrheit	
		Positiv	**Negativ**
Klassifikation	**Positiv**	TP	FP
	Negativ	FN	TN

Tabelle 4.1: Konfusionsmatrix eines binären Klassifizierers

Wie auch in Tabelle 4.1 zu sehen, gilt:

- TP: Der vorhergesagte Wert und der Referenzwert sind beide positiv.
- TN: Der vorhergesagte Wert und der Referenzwert sind beide negativ.
- FP: Der vorhergesagte Wert ist positiv und der Referenzwert ist negativ.
- FN: Der vorhergesagte Wert ist negativ und der Referenzwert ist positiv.

Für die Berechnung der folgenden Maße wird jeweils die Anzahl der Beispiele in den Kategorien TP, FP, FN und TN gezählt [31].

4.4.2 Genauigkeit

Für die Berechnung der Genauigkeit (engl. Accuracy) wird der Anteil der korrekt vorhergesagten Datenpunkte bestimmt [31].

$$\text{Genauigkeit} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.4}$$

Die Genauigkeit lässt sich auch klassenunabhängig für die Gesamtvorhersage berechnen, indem die Zahl der korrekt klassifizierten Datenpunkte durch die Gesamtzahl der Datenpunkte geteilt wird [32]. Die Genauigkeit über alle Klassen kann jedoch durch unbalancierte Datensätze stark verzerrt werden.

4.4.3 Sensitivität

Die Sensitivität - auch Recall genannt - gibt den Anteil der korrekt positiv erkannten Datenpunkte einer Klasse an den Datenpunkten der Klasse an [31], also wie sicher eine Klasse erkannt wird, wenn sie vorliegt.

$$\text{Sensitivität} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.5}$$

4.4.4 Spezifität

Die Spezifität (engl. specificity) gibt an, wie viele von allen tatsächlich negativen Werten korrekt als negativ vorhergesagt werden [31], also wie sehr man sich darauf verlassen kann, dass ein negativ vorhergesagter Datenpunkt tatsächlich negativ ist.

$$\text{Spezifität} = \frac{\text{TN}}{\text{FP} + \text{TN}} \tag{4.6}$$

4.4.5 Präzision

Die Präzision gibt an, wie viele von allen positiv vorhergesagten Werten korrekt als positiv vorhergesagt wurden. Sie kann also als Gegenteil der Sensitivität angesehen werden und gibt an, wie sicher die Hypothese nicht erfüllt ist, wenn sie nicht vorhergesagt wurde [31].

$$\text{Präzision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.7}$$

4.4.6 F_1-Score

Der F1-Score bezieht sowohl die Präzision als auch die Sensitivität mit ein und bildet damit ein Kriterium, in dem beide einbezogen sind. Für binäre Klassifikationen ist er wie folgt zu berechnen:

$$F_1 = \frac{2 \cdot \text{Präzision} \cdot \text{Sensitivität}}{\text{Präzision} + \text{Sensitivität}} \tag{4.8}$$

Der Wert des F_1-Scores liegt also immer zwischen beiden Werten, ist jedoch näher am niedrigeren von beiden [31, 33].
Der F_1-Score kann stattdessen auch über

$$\frac{2 \cdot |X \cap Y|}{|X| + |Y|} \tag{4.9}$$

berechnet werden und wird dann oft Dice-Koeffizient genannt. X bezeichnet dabei die Referenzdaten und Y die Vorhersage durch das Modell. Beide sind binäre Vektoren und stehen für das Auftreten einer Klasse in den Daten. Diese Formel kann so angepasst werden, dass binäre Referenzdaten mit kontinuierlichen Wahrscheinlichkeiten als Vorhersage verglichen werden können [34].

4.4.7 Konfusionsmatrix

Die Konfusionsmatrix gibt nicht direkt einen Wert aus, anhand dessen Modelle verglichen werden können. Dafür kann an ihr abgelesen werden, welche Klassen von einem Modell miteinander verwechselt werden.
Konfusionsmatrizen sind wie in Tabelle 4.1 aufgebaut, können aber auch für mehrere Klassen erstellt werden. Dazu bekommt jede Klasse eine Spalte und eine Zeile, wobei die Klassen in beiden gleich geordnet sein sollten. Eine von beiden steht für die wahren Klassen der Referenzdaten und die andere für die vorhergesagten Klassen. So kann jeder Datenpunkt in ein Feld der Matrix einsortiert werden und die Zellen geben für den verwendeten Datensatz die Anzahl der Datenpunkte an, die in der Referenz-Vorhersage-Kombination aufgetreten sind. Wenn nichts falsch klassifiziert wurde, stehen also außerhalb der Diagonale von links oben nach rechts unten nur Nullen [35].

4.5 Erklärbarkeit und Interpretierbarkeit

4.5.1 Übersicht

Einfache Modelle wie Entscheidungsbäume haben den Vorteil, dass sich ihre Entscheidungen durch den Menschen leicht nachvollziehen lassen: Es kann überprüft

werden, was das Modell gelernt hat und wie es seine Entscheidungen trifft. Diese White-Box-Modelle werden auch interpretierbare Modelle genannt. Ein Nachteil dieser Modelle ist, dass sie nicht sehr komplex und daher auch weniger leistungsfähig sind. Das wiederum führt zu schlechteren Ergebnissen bei komplexeren Aufgaben. Black-Box-Modelle hingegen sind deutlich komplexer, dadurch aber auch nicht mehr interpretierbar.
Die erklärbare künstliche Intelligenz (auch XAI für explainable artificial intelligence) beschreibt Methoden, die es Menschen ermöglicht, annäherungsweise Erklärungen für die Funktionsweise der Modelle zu finden und wesentliche Entscheidungsgrundlagen nachzuvollziehen. Das Ziel dabei ist, das Modell näher zu verstehen, um für Transparenz bei der Entscheidungsfindung zu sorgen und so das Vertrauen in die Modelle zu stärken.
Mit erklärbarer KI können unterschiedliche Fragestellungen beantwortet werden: Hat das Modell gelernt was es sollte, bzw. was erwartet wurde? Warum klassifiziert ein Modell einige Beispiele falsch? Kann aus dem Modell Wissen über die Daten oder die Anwendung generiert werden?
Zur Beantwortung dieser Fragen existieren unterschiedliche Algorithmen, die sich in Kategorien einteilen lassen: Es gibt lokale Erklärungsmethoden, die Informationen zur Bewertung einzelner Beispiele liefern und globale Algorithmen, die eine Gesamtbeschreibung liefern, wie das Modell funktioniert. Außerdem gibt es sowohl Methoden, die nur für spezifische Modelle entwickelt wurden als auch Algorithmen, die unabhängig von dem verwendeten Modell funktionieren (modell-agnostische Methoden). Modell-agnostische Algorithmen arbeiten ohne jegliches Wissen über das Modell. Stattdessen variieren sie die Eingabedaten und werten die Veränderung der Ausgabe aus [36].

4.5.2 Permutation Feature Importance

Die Permutation Feature Importance zählt zu den globalen modell-agnostischen Methoden und liefert einen sehr komprimierten Einblick in Modelle. Für den Algorithmus werden nacheinander die Werte aller Beispiele eines Features untereinander getauscht und die Daten durch das Modell bewertet. Der Loss wird gemessen und gibt an, wie groß der Einfluss des Features auf den Fehler ist. Wenn der Fehler gering ist, ist das Feature nicht bedeutsam, da eine Veränderung nicht zu einer Änderung des Ergebnisses führt. Ist die Änderung jedoch größer, so wird angenommen, dass das Feature wichtig für das Modell ist.
Um die Permutation Feature Importance zu berechnen, wird zunächst der Fehler der Vorhersage auf den originalen Daten errechnet. Dafür wird ein annotierter Datensatz benötigt, der nicht für das Training verwendet wurde, da dies das Ergebnis beeinflussen kann. Die Feature-Importance kann durch den Quotienten des veränderten Losses mit dem originalen Loss oder deren Differenz errechnet werden.
Die Permutation Feature Importance berücksichtigt Interaktionen zwischen Features. Das hat allerdings den Nachteil, dass korrelierte Features weniger wichtig wirken können, da der Loss weniger niedrig ist, wenn das Modell stattdessen einen zusammenhängenden Wert verwendet [36].

4.5.3 Counterfactual Explanations

Counterfactual Explanations sind ein lokales, also beispielbasiertes, modell-agnostisches Verfahren. Es gibt jedoch auch modell-spezifische Implementierungen von Counterfactual Explanations. Sie geben an, welche kleinsten Veränderungen an einem

Datenpunkt gemacht werden müssen, damit sich die Vorhersage dieses Datums zu einer bestimmten Klasse ändert. Damit ermöglichen sie zum Beispiel das Debugging bei der falschen Einordnung von Daten. Es sind jedoch mehrere Counterfacturals pro Beispiel möglich. Ein Vorteil von Counterfactual Explanations ist, dass sie sich auch ohne Daten berechnen lassen, sodass ohne jegliches Wissen über die Domäne Einblicke gewonnen werden können.
Counterfactual Explanations werden meist per Optimierungsalgorithmen bestimmt. Es wird ein Datenpunkt aus den Daten ausgesucht oder generiert und eine zu erreichende Klasse festgelegt. Eine Lossfunktion definiert die Ähnlichkeit des ausgewählten Beispiels zu dem Counterfactual. Die Lossfunktion berücksichtigt dazu die Klassen der Predictions, die Ähnlichkeit der Features und ist in der Lage weitere Punkte, wie die Anzahl der Veränderungen mit einzubeziehen. Ein zufälliges Beispiel wird generiert und per Loss-Minimierung an das Ziel angeglichen. Das Ergebnis ist ein neuer Datenpunkt, der möglichst ähnlich zu dem Ausgangsdatum ist, jedoch durch das Modell zur Zielklasse zugeordnet wird [36].

4.5.4 Activation Maximization

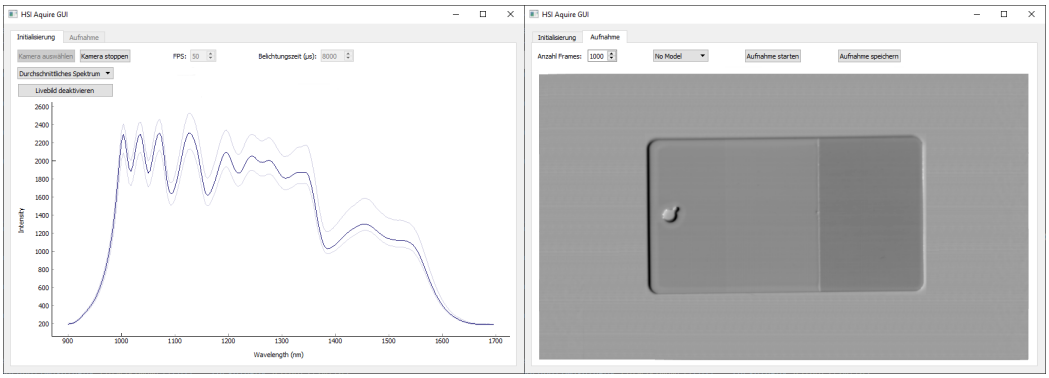
Activation Maximization ist ein Verfahren, mit dem die gelernten Features einer beliebigen Schicht eines neuronalen Netzes visualisert werden können. Es handelt sich dabei um einen modell-spezifischen, globalen Algorithmus.
Die Idee der Activation Maximization ist es, die Eingabedaten oder eine Feature-Map so zu gestalten, dass die Aktivierung eines bestimmten Ausgabeneurons maximiert wird. So kann visualisiert werden, wie sich ein Netz die Eingabedaten oder eine Feature-Map einer bestimmten Klasse optimalerweise *vorstellt*.
Dafür wird ein Neuron der Ausgabeschicht stark aktiviert und die Auswirkung per Backpropagation auf die Eingabe zurückgerechnet. Im Folgenden werden nicht die Gewichte des Netzes entsprechend der Gradienten angepasst, sondern die Eingabe selbst. So wird die Eingabe so optimiert, dass das ausgesuchte Neuron möglichst stark aktiviert wird. Dabei ist jedoch zu beachten, dass die Aktivierung vor einer eventuellen Softmax-Aktivierung in der Ausgabeschicht erfolgen muss, da die Gradienten sonst zu klein werden.
Die verwendete Eingabe - also der Input oder die Feature-Map, die optimiert werden soll - kann initial zum Beispiel aus zufälligem Rauschen bestehen.
Durch gestufte Faltungen und Pooling-Operationen entsteht bei der Optimierung jedoch hochfrequentes Rauschen. Um dieses zu verhindern, sind Regularisierungen während der Optimierung nötig, die zum Beispiel hohen Frequenzen in der Eingabe entgegen wirken [37].

5 Umsetzung

5.1 Datensatz

5.1.1 Aufnahme

Für die Aufnahme der Hyperspektralbilder wurde ein Tool mit Python implementiert. Ein Vorteil davon ist, dass keine unbekannte Vorverarbeitung der Daten außerhalb der Kamera geschieht, sodass der Prozess nachvollziehbarer bleibt. Die Existenz einer funktionierenden Schnittstelle der Kamera für die spätere Inline-Verwendung der Software ist ein weiterer Vorteil.
Für die Schnittstelle zur Kamera wird die Python-Bibliothek harvesters verwendet, die vom GenICam Commitee entwickelt wird. Harvesters nutzt zur Kommunikation mit der Kamera Bildaufnahmebibliotheken, die GenTL-Producer genannt werden. Die dazugehörige GenTL-Datei, die für diese Anwendung verwendet wird, wurde mit Software von Stemmer Imaging erzeugt.
Das Tool kann so auf die Kamera zugreifen, Parameter einstellen und Frames aufnehmen. Die GUI besteht aus zwei Tabs: Der erste Tab dient zur Initialisierung der Kamera und ihrer Einstellungen und der zweite zur Aufnahme von Hyperspektralbildern.
Im Einstellungstab (siehe Abbildung 5.1a) lässt sich eine GenTL-Datei auswählen und die Framerate und Beleuchtungszeit der Kamera einstellen. Außerdem kann eine Liveansicht aktiviert werden, die wahlweise die Intensität über die Zeilenbreite oder das durchschnittliche Spektrum der aufgenommenen Zeile anzeigen kann. Mit der ersten der beiden Ansichten kann die Linse der Kamera eingestellt werden. Dazu muss für einen starken Kontrast in der aufgenommenen Zeile gesorgt werden. Dies kann zum Beispiel durch eine schwarze Pappe auf der Kalibrierungsreferenz geschehen, sodass die Kamerazeile beides aufnimmt. Wenn die Flanke zwischen hell und dunkel sehr steil wird, ist die Linse scharf eingestellt.
Im zweiten Tab (siehe Abbildung 5.1b) kann die Anzahl der Frames, die aufgenommen werden sollen, eingestellt werden und die Aufnahme gestartet werden. Das aufgenommene Bild wird als Intensität über die räumlichen Dimensionen angezeigt und kann gespeichert werden.
Sobald die Kamera gestartet wurde, läuft die Verwertung der eingehenden Frames



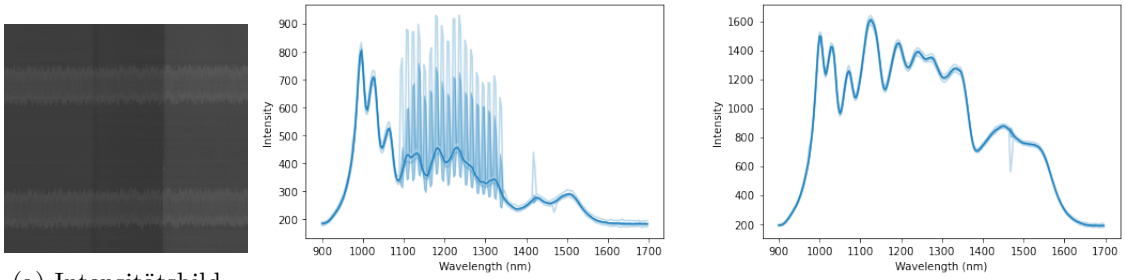
(a) Einstellungstab (b) Aufnahmetab

Abbildung 5.1: Tabs des Tools zur Aufnahme von Hyperspektralbildern

in einem eigenen Thread. Wenn eine Zeile aufgenommen wurde, sendet der Thread ein Signal mit dem Frame und seiner Identifikationsnummer. Diese sind aufsteigend, sodass die Frames nachfolgender IDs zu einem Bild zusammengesetzt werden können. Zwar wäre es mit Harvester auch möglich, weitere Einstellungen vorzunehmen, jedoch beschränkt sich die Auswahl in der GUI auf die am häufigsten genutzten. Für komplexere Einstellungen wird für diese Arbeit Perception Studio verwendet. So wurde zum Beispiel auch eingestellt, dass die Kamera selbst das Bild mit der Smile-Korrektur verbessert.
Für die Aufnahme der Bilder wurde eine Framerate von 50 Frames per Second (FPS) ausgewählt. Die Kamera fährt in einer linearen Bewegung im Abstand von ca. $186mm$ über die Probe. Dadurch ergibt sich eine Auflösung in Zeilenrichtung von $0.2\frac{mm}{px}$. Der Roboter muss also mit $0.01\frac{m}{s}$ fahren, um ebenfalls eine Auflösung $0.2\frac{mm}{px}$ in Verfahrrichtung zu erreichen. Die Belichtungszeit wurde mit $8ms$ so gewählt, dass eine von der Reflektionsreferenz aufgenommene Zeile gut, aber nicht überbelichtet wird. Dazu wird ein Bild der Reflektionsreferenz aufgenommen und die Belichtungszeit so angepasst, dass die gemessene Intensität im Bereich der maximal möglichen Werte aber nicht darüber liegt.
Für je ein Bild werden etwa 1000 Zeilen aufgenommen. So ist zwar etwas Platz an den Seiten, dafür kann der Moment besser abgeschätzt werden, wann die Aufnahme gestartet werden muss. Damit dieses noch vereinfacht wird, wird die Pilot-Linie der Beleuchtung aktiviert, die den beleuchteten Bereich zusätzlich mit sichtbarem Licht markiert. Da die Beleuchtung auf die Kamera eingestellt ist, kann so sichtbar gemacht werden, wo in etwa die Kamera gerade eine Zeile aufnimmt. Bei der NIR-Beleuchtung wurde das Beleuchtungsprofil mit der höchsten Leistung der LEDs ausgewählt. Dadurch ist der gesamte Bereich möglichst homogen beleuchtet und das Rauschen wird reduziert. Wie die Bilder aussehen können, wenn ein anderes Beleuchtungsprofil, das die Leistung der LEDs reduziert, verwendet wird, ist in Abbildung 5.2 zu sehen. Erste Aufnahmen wurden mit diesem Beleuchtungsprofil gemacht. Diese zeigten jedoch teilweise - sowohl in räumlicher als auch in spektraler Dimension - starkes Rauschen, sodass das Beleuchtungsprofil mit maximaler Intensität ausgewählt wurde.

5.1.2 Annotierung

Da die Daten segmentiert, also pixelweise klassifiziert werden sollen, ist auch eine pixelweise Annotation der Trainingsdaten nötig. Dazu wird pro Aufnahmebild ein



(a) Intensitätsbild (b) Spektrum im verrauschen Bereich (c) Spektrum Beleuchtungsprofil 2

Abbildung 5.2: Die Daten des Bilds 5.2a und des Spektrums 5.2b wurden mit Beleuchtungsprofil 1 aufgenommen. Deutlich zu sehen ist der räumlich (5.2a) und spektral (5.2a, Standardabweichung in Mittelblau, Extremwerte in hellblau) begrenzte Bereich des Rauschens. Das Spektrum in 5.2c wurde mit dem zweiten Beleuchtungsprofil aufgenommen. Es ist zu erkennen, dass das Bild besser beleuchtet wurde und deutlich weniger Rauschen enthält.

PNG erstellt, welches 8-bit-Werte enthält. Jeder dieser Werte steht für eine Klasse. Für die Abbildung, welche Zahl welche Klasse repräsentiert, existiert eine Liste. Diese Liste speichert pro Klasse neben der ID und der Bezeichnung des Lacks noch weitere Metadaten und eine RGB-Tupel. Letzteres speichert die Farbe mit der der Lack in klassifizierten oder annotierten Bildern dargestellt wird. Als Metadaten werden pro Klasse der Hersteller des Lacks, die Angabe der Lackart (Primer, Basecoat, etc.), die Anzahl der Laserzyklen und bei Klarlacken der Lack in der darunter liegenden Schicht gespeichert. Mit diesen Metadaten lassen sich mehrere Klassen auch im Nachhinein automatisiert zusammenfassen oder Zusammenhänge analysieren.
Die ID 0 steht für undefinierte Klassen und wird verwendet, um Bereiche bei der Annotation auszulassen, auf denen nicht trainiert wird. Alle weiteren Klassen sind vorkommende Lacke oder die aufgenommenen Hintergründe Alumnium oder Reflektionsreferenz. Durch vielfältige Kombinationsmöglichkeiten der Lacke und Bearbeitungen - zum Beispiel, dass jeder Lack je einmal unbearbeitet und bis zu vierfach gelasert werden kann - entstehen mehr als 130 Klassen.
Die Verwendung von existierender Software zur Annotierung der Bilder wie der Matlab Image Labeler wurde durch mehrere Punkte erschwert: Ein Problem ist, dass die Hyperspektralbilder zunächst in gängige Bildformate konvertiert werden müssen, ehe sie in einer dieser Softwares annotiert werden können. Außerdem sollte die Klassendefinition aufgrund der Vielzahl an Klassen inklusive der Metadaten automatisiert erzeugt werden können und die Auswahl der Klassen beim Annotieren übersichtlich gestaltet sein. Da keine Software gefunden werden konnte, die diese Punkte zufriedenstellend erfüllt, wurde ein eigenes Annotierungstool (siehe Abbildung 5.3) für die speziellen Anforderungen erweitert. So war es möglich, dass die Klassendefinition als JSON-Datei aus den möglichen Kombinationen automatisiert erzeugt wurde. Die Farben für die Darstellung der Klassen wurden zufällig erzeugt, sodass keine Farbe zweimal vorhanden ist. Aufgrund der Vielzahl sind jedoch sehr ähnliche Farben möglich.
Das Tool ist mit Python geschrieben und enthält eine Qt-GUI. Es kann ein Ordner ausgewählt werden, aus dem die Bilder annotiert werden. Diese Bilder können wie in einer Fotogalerie nacheinander geöffnet und dann wie in einem Grafikprogramm

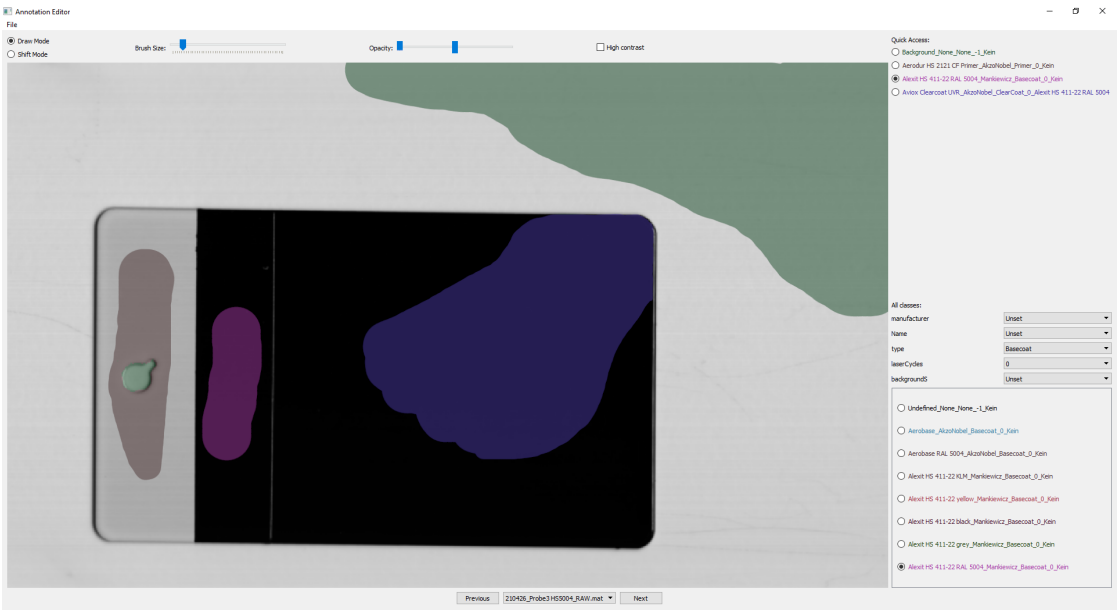
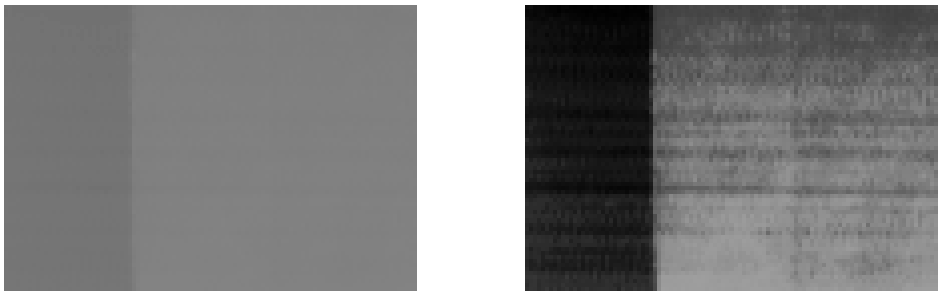


Abbildung 5.3: Tool zur Annotierung der Proben mit teils gelabeltem Intensitätsbild

übermalt werden. Die Hyperspektraldaten werden als Intensitätsbild angezeigt. Darüber wird halbdurchsichtig die Annotierung mit der Farbe der jeweiligen Klassen angezeigt. Die Annotierung erfolgt analog zum Zeichnen in einem Grafikprogramm: Statt einer Farbe wird die Klasse ausgewählt. Um die betreffende Klasse schnell zu finden, können Klassen nach ihren Eigenschaften wie dem Typ des Lacks oder der Anzahl der Laserzyklen gefiltert werden. Alle für ein Bild verwendeten Lacke sind zudem im Schnellzugriff verfügbar. Die Größe des Pinsels ist verstellbar, um große Bereiche schnell und grob und feine Bereiche ausreichend kleinteilig annotieren zu können. Dazu ist auch ein Zoom ins Bild möglich. Die Zuordnung der Aufnahmen zu den Annotationen erfolgt über den gleichen Dateinamen.
Die vorhandenen Aufnahmen wurden so gut wie möglich annotiert, allerdings existieren dabei einige Schwierigkeiten: Bei einigen der Bilder lassen sich unterschiedliche Lacke oder Zustände anhand des Intensitätsbildes nicht unterscheiden. Dies kann teilweise dadurch verbessert werden, dass der Kontrast des Bildes erhöht wird (siehe Abbildung 5.4). Bei sehr dunklen Bildern führte das jedoch nicht zum Erfolg. In diesem Fall wurden nur die äußeren Bereiche annotiert und die Fläche, auf der die beiden Lacke zusammenstoßen aufgrund der Unsicherheit ausgelassen. Zudem war ein Problem, dass feine Strukturen, sehr aufwendig zu annotieren sind. Daher wurden diese - wenn bereits genügend Trainingsdaten dieser Klasse vorhanden waren - ebenfalls ausgelassen. Eine weitere Schwierigkeit bei der Annotation sind Übergangsbereiche, bei denen die Klassenzugehörigkeit auf dem Intensitätsbild und teilweise auch auf der Probe schwierig zu bestimmen sind. Aufgrund dieser Probleme sind einige falsch annotierte Pixel zu erwarten.

5.1.3 Datensatzerstellung

Für die Datensatzerstellung stehen 23 annotierte Bilder von 21 Probenblechen zur Verfügung, auf denen alle zu erkennenden Lacke sowie das Aluminium des Probenblechs und der Keramikhintergrund zu sehen sind. Viele Lacke bzw. Lackkombinationen sind nur auf jeweils einer Aufnahme abgebildet.



(a) Normaler Kontrast (b) Hoher Kontrast

Abbildung 5.4: Beide Bilder zeigen den gleichen Bereich einer Probe, auf der zu jeweils etwa einem Drittel unterschiedliche Lacke aufgetragen sind. Während bei normalem Kontrast der mittlere nicht von dem rechten Lack zu unterscheiden ist, ist bei erhöhtem Kontrast eine Trennlinie zu erahnen.

Um unabhängiger von der Beleuchtung zu werden und die Daten für das Training mit den neuronalen Netzen vorzubereiten, werden sie nach der Formel 2.1 normiert. Dazu wurde pro Aufnahmesession ein Weißabgleichsbild und ein Dunkelbild erstellt. Der Datensatz muss in Trainings-, Validierungs- und Testdaten aufgeteilt werden, um die Modelle zu erstellen, zu trainieren und zu evaluieren. Diese Datensätze sollten möglichst unabhängig voneinander sein. Außerdem ist für den Datensatz wichtig, dass er noch räumliche Informationen enthält, da auch Modelle ausgewählt wurden, die zusätzlich zu den spektralen Informationen die räumlichen verwenden. Außerdem wird erwartet, dass benachbarte Pixel sehr ähnlich sind, was der Unabhängigkeit der Datensätze untereinander entgegensteht. Daher werden nicht die Pixel zufällig aufgeteilt, sondern die Bilder in Kacheln eingeteilt. Als Größe für die Kachel wurde 32^*40px gewählt. Damit nicht zufällig auf dem Übergang zwischen zwei Lacken auch immer eine Grenze benachbarter Kacheln erscheint, werden die Kacheln zeilenweise verschoben.
Die Kacheln der Aufnahmen bzw. der Annotationen werden jeweils einzeln und als Numpy-Array bzw. PNG gespeichert. Als Dateiname wird ein Universally Unique Identifier (UUID) verwendet, um die Daten und die Annotationen zuordnen zu können. Außerdem existiert eine Liste, die es anhand der UUID erlaubt, die Kacheln zu ihrer Position im Originalbild zuordnen zu können.
Da einige Bereiche auf den Bildern nicht eindeutig annotiert werden konnten, haben einige Kacheln nur Pixel mit undefinierten Klassen. Diese Kacheln werden für das Training und die Evaluierung ebenso wie die Kacheln an den Rändern, die kleiner sind als die definierte Größe, aussortiert. Außerdem ist die Keramik-Hintergrund-Klasse in dem Datensatz deutlich überrepräsentiert. Daher werden nur wenige Kacheln ausgewählt, auf denen ausschließlich Hintergrund abgebildet ist. Insgesamt werden so 9576 Kacheln aussortiert. Die übrig bleibenden Kacheln werden zufällig zu 80% in den Trainingsdatensatz und zu je 10% in die Validierungs- und Testdaten aufgeteilt. Der resultierende Trainingsdatensatz besteht aus 4120 Kacheln, der Validierungsdatensatz aus 515 Kacheln und der Testdatensatz aus 516 Kacheln. Alle Datensätze haben ähnliche Verteilungen der Klassen bei denen sich das relative Vorkommen der einzelnen Klassen um maximal 2% unterscheidet.
Betrachtet man den Datensatz mit allen Klassen unter Berücksichtigung der Anzahl der Laserzyklen, enthält der Datensatz 81 Klassen. Wird nur zwischen gelasert und

ungelasert unterschieden, reduzieren sich die Klassen auf 41 und bei Unterscheidung nur nach aufgetragenem Lack und unabhängig von der Laserbehandlung lassen sich die Daten in 25 Klassen unterteilen.
Welche Klassen existieren und wie viele Beispiele jeweils existieren lässt sich in Abbildung 5.5 ablesen. Im ersten Ring sind die Klassen nach den Lacken - bzw. bei den Klarlacken auch nach den darunter aufgetragenen Basislacken - aufgeteilt. Die Größe der Segmente steht dabei für die Anzahl der zugehörigen Beispiele im Datensatz. Der äußere Ring und damit auch die Farbe gibt an, wie viele der jeweiligen Beispiele der Klasse wie oft gelasert wurden. Es ist zu erkennen, dass von einigen Lacken insgesamt nur wenige Beispiele existieren und von einigen Lacken keine gelaserten Datenpunkte vorhanden sind.

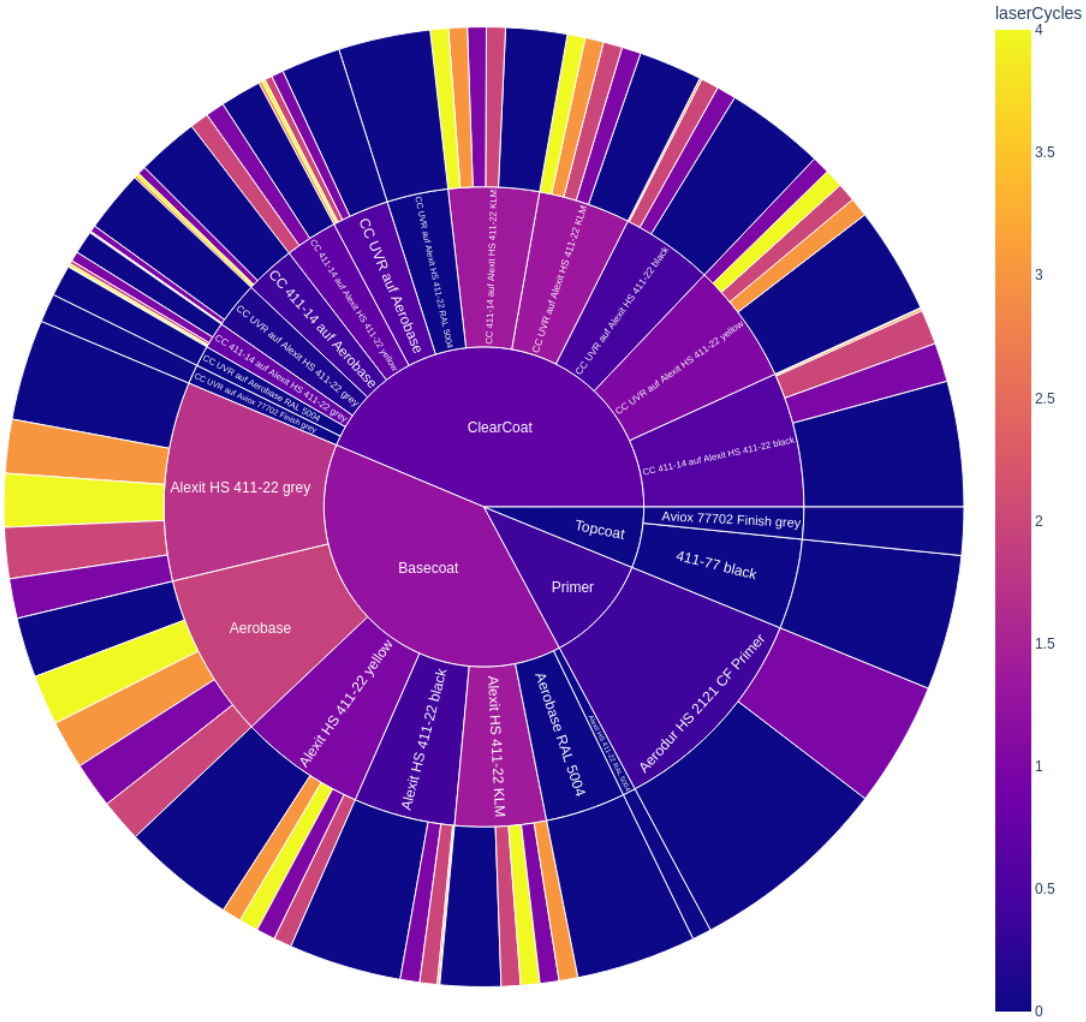


Abbildung 5.5: Verteilung der Klassen im Datensatz

5.2 Random Forest

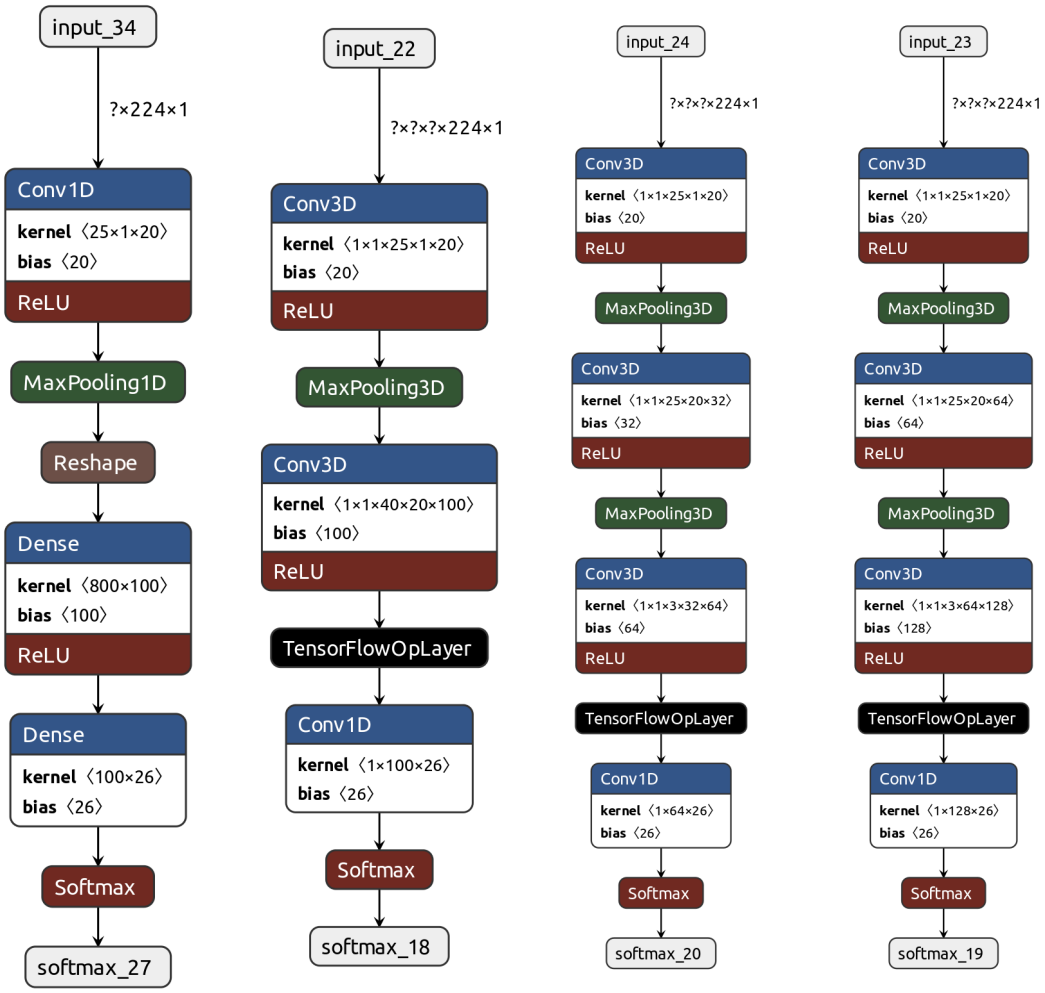
Für das Training der Random Forest Modelle wird der `RandomForestClassifier` von scikit-learn verwendet. Bei diesem können unter anderem die Anzahl der Bäume und die maximale Anzahl der Blätter festgelegt werden. Es werden jeweils mehrere Modelle für die Zielklassen *Lack inklusive Laserbearbeitungszyklen*, *Lack gelasert/ungelasert*

und *Lack unabhängig der Laserbearbeitung* trainiert und dabei die festgelegten Hyperparameter variiert. Die Variationen der Anzahl der Bäume sind dabei für alle Zielklassen gleich. Die Trainings werden jeweils mit 10, 50 und 100 Bäumen durchgeführt. Die Anzahl der maximalen Blätter hingegen wird pro Zielklasse gewählt und entspricht mindestens der Anzahl der jeweiligen Klassen, um die Erkennung jeder Klasse zu ermöglichen. Außer bei der Erkennung aller Klassen wird die maximale Anzahl der Blätter in acht Schritten bis zur achtfachen Klassenanzahl erhöht. Da das bei allen Klassen 640 Blättern entsprechen und das Training dafür sehr lange dauern würde, wird als maximaler Parameter hier das Vierfache der Klassen gewählt.
Alle Modelle erhalten als Eingabe jeweils das Spektrum eines Pixels und berechnen daraus die Klasse.
Außerdem werden noch Random Forests trainiert, die zusätzlich die benachbarten Spektren erhalten, um insbesondere bei Lackübergängen weitere Informationen bereitstellen zu können. Diese erhalten nicht nur das Spektrum eines Pixels sondern noch die der acht Nachbarn. Zu diesen werden jeweils durch einen Random Forest die Wahrscheinlichkeiten des Auftretens der Klassen errechnet. Diese Ergebnisse werden zu einem Vektor konkateniert, welcher den Input für einen weiteren Random Forest bildet und die Klasse des mittleren Pixels bestimmt. Als Basismodelle werden nach dem Training der rein spektralen Random Forests einige davon ausgewählt. Der Random Forest der zweiten Stufe wird neu trainiert. Die Anzahl der Bäume bleibt dabei konstant, während die maximale Anzahl der Blätter wieder variiert.
Für das Training werden die Klassenwahrscheinlichkeiten zunächst für jedes Pixel eines Bildes mit dem Basismodell errechnet und in dreidimensionalen Matrizen der Größe $w \times h \times n_{classes}$ (w: Breite des Bilds, h: Höhe des Bildes, $n_{classes}$: Anzahl der Klassen) gespeichert. Diese Matrizen werden in der räumlichen Dimension gepaddet, indem die Spektren der Randpixel kopiert werden. Dadurch lassen sich auch die Randpixel klassifizieren. Diese Matrix wird in 3×3 Kacheln, die jeweils um eins verschoben werden, aufgeteilt. Da das Modell nicht lernen soll, die Undefined-Klasse vorherzusagen, werden die Kacheln, bei der die Klasse des mittleren Pixels unbekannt ist, aussortiert. Jede dieser übrig bleibenden Kacheln ist ein Trainingsdatum für den Random Forest in der zweiten Stufe.

5.3 Neuronale Netze

Die neuronalen Netze werden mit Keras unter der Verwendung der Grafikkarte erstellt und trainiert. Für die Suche nach einer geeigneten Architektur werden für die unterschiedlichen Zielklassen je vier Netze trainiert. Die Architektur von Modell A orientiert sich dabei direkt an der Struktur aus dem in Kapitel 2.3 vorgestellten Paper. Diese hat jedoch den Nachteil, dass jeweils nur genau ein Spektrum verarbeitet werden kann. Das hat den Grund, dass sich die vom Netz zusammengefassten Informationen vor der vollständig verbundenen Schicht sowohl auf der Dimension der Spektren als auch der Dimension der Faltungskanäle befinden. Dadurch wird in Keras ein Reshaping nötig, für das die Größe konkret angegeben werden muss. Selbst wenn der Dense-Layer durch eine äquivalente Faltungsschicht ersetzt werden würde, könnte deshalb nicht der Vorteil von Faltungsnetzen, dass die Eingaben nahezu beliebige Eingabegrößen haben können, genutzt werden. In der Praxis kann dies zu längeren Ausführungszeiten beim Klassifizieren neuer Daten führen. Aus diesem Grund werden zusätzlich drei Architekturen verwendet, die mit beliebigen Bildgößen funktionieren. Lediglich die Anzahl der Kanäle des Eingabebildes ist festgelegt. Diese

Architekturen sind in Abbildung 5.6 dargestellt.
Bereits die Eingabeschichten der Modelle B, C und D unterscheiden sich von Modell



(a) Modell A (b) Modell B (c) Modell C (d) Modell D

Abbildung 5.6: Architekturen der verwendeten Netze

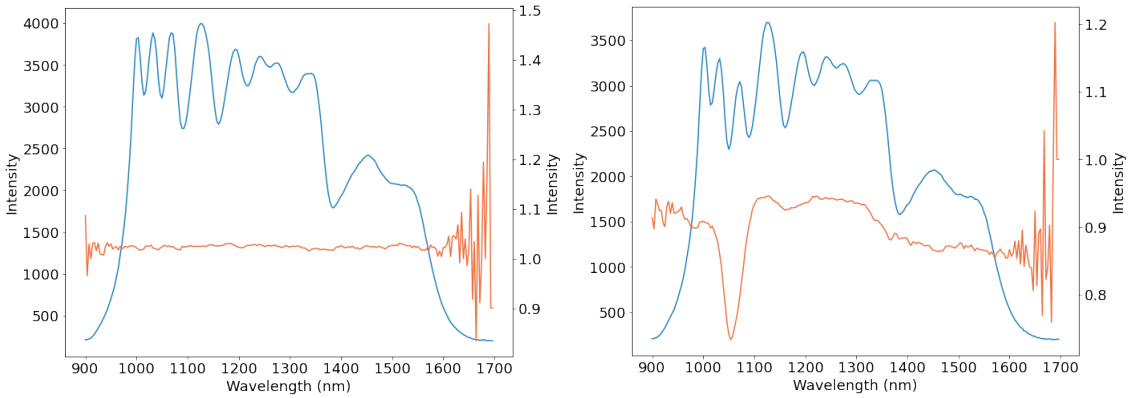
A. Während bei Modell A genau ein Spektrum verarbeitet werden kann, ist die Eingabe für die anderen Modelle fünfdimensional, sodass ein Bild mit mehreren Spektren verarbeitet werden kann. Da die Größe des Bildes nicht angegeben ist, kann es beliebig groß sein. Dementsprechend sind auch die Faltungen nicht mehr eindimensional, sondern dreidimensional. Für die Netze, die jedes Spektrum ausschließlich einzeln betrachten, findet die Faltung jedoch nur auf der Dimension der Spektren statt.
Modell B ist eine zu Modell A sehr ähnliche Version, die statt der ersten vollständig verbundenen Schicht eine weitere Faltungsschicht besitzt, deren Faltungskernel so gewählt ist, dass die Größe der Dimension der Spektren auf Eins reduziert und stattdessen die Anzahl der Kanäle auf 100 erhöht wird. Die Dimension der Spektren kann so ohne Informationsverlust reduziert werden. Die letzte Faltungsschicht entspricht so der zweiten vollständig verbundenen Schicht aus Modell A.
Während Modell A die Eignung der in verwandten Forschungen vorgeschlagenen Architektur für die hier betrachtete Anwendung überprüfen soll und Modell B versucht, diese Architektur für Bilder beliebiger Größe anzupassen, soll mit Modell C und D

analysiert werden, ob tiefere Netze in dem vorliegenden Anwendungsfall Vorteile bringen können.
Modell C ist dazu eine Variante von Modell B, die statt dem zweiten breiten Faltungskernel zwei Faltungsschichten mit kleineren Kernels und eine Pooling-Schicht enthält. Modell D entspricht Modell C, nur dass die Anzahl der Kanäle in der zweiten und dritten Faltungsschicht verdoppelt wurden.
Die in 5.6 abgebildeten Architekturen stammen von den Netzen, die ausschließlich zur Erkennung der Lacke trainiert wurden. Die Netze zur Unterscheidung der Klassen in gelasert und ungelasert entsprechen diesen Architekturen bis auf die Anzahl der Ausgabeneuronen, die an die Anzahl dieser Zielklassen angepasst wurden. Gleiches gilt für die Zielklassen, die die Laserzyklen beinhalten, für Modell A und B. Da jedoch insbesondere bei Modell C mehr Neuronen auf der Ausgabeschicht als auf der vorherigen Schicht vorhanden sein müssten, wurden hier die Anzahl der Kanäle von Modell B und C in der zweiten und dritten Faltungsschicht verdoppelt.
Auch bei den neuronalen Netzen wird analysiert, ob die Betrachtung der benachbarten Spektren zu einer Verbesserung der Vorhersage führen kann. Modell A eignet sich jedoch aufgrund der bereits erläuterten Probleme nicht ohne größere Veränderungen für diese Analysen. Stattdessen wurden Modell B und C dafür ausgewählt. Da bei neuronalen Netzen der Zeitpunkt für die Faltung über die räumliche Dimension frei gewählt werden kann, wird hier untersucht, ob sich ein Aggregieren der Rohdaten oder ein Zusammenfassen der bereits stark verarbeiteten Daten positiver auswirkt. Dazu werden in Modell B und Modell C jeweils eine zusätzliche Faltungsschicht vor die erste oder vor die letzte Faltungsschicht eingefügt. Diese Faltungsschicht besitzt einen $3 * 3 * 1$-Kernel und 20 Kanäle für die anfängliche räumliche Faltung bzw. die Anzahl der Kanäle der Vorgängerschicht für die späte räumliche Faltung.
Für das Training dieser Modelle werden die Bilder mit einem Pixel, der die Spektren der Randpixel kopiert, in den räumlichen Dimensionen gepaddet.
Durch die hohe Anzahl der Kanäle sind die Hypercubes sehr speicherintensiv. Daher wird für das Training ein eigener Datengenerator verwendet, der zudem das Padding übernimmt. Außerdem werden per Callback-Funktion während des Trainings die besten Modelle gespeichert, sodass im Fall von Overfitting ein Modell einer früheren Epoche ausgewählt werden kann.
Eine weitere Besonderheit beim Training ist das Vorhandensein der Undefined-Klasse. Dadurch, dass die Daten nicht wie beim Random Forest in einzelne Trainingsdaten aufgeteilt werden, sondern ganze Bilder auf einmal bearbeitet werden, können die Pixel, deren Klasse unbekannt ist, nicht vor dem Training aussortiert werden. Daher erhält das Modell ein Ausgabeneuron, welches *Undefined* vorhersagen könnte, es wird aber so trainiert, dass das Modell diese Klasse ignoriert. Dazu wird als Lossfunktion eine gewichtete Kreuzentropie verwendet: Die Gewichte sind für alle Klassen Eins und nur für die Undefined-Klasse Null. Dadurch gehen die Pixel, zu denen keine Klasse zugeordnet wurde bzw. werden konnte, nicht mit in das Training mit ein.

6 Analysen

6.1 Spektren

Um die Probleme der Modelle besser verstehen und unter Umständen Schwierigkeiten entgegen wirken zu können, ist es hilfreich die Daten genauer zu verstehen und mögliche Ähnlichkeiten zwischen Klassen aufzudecken.
In Abbildung 6.1 ist dazu zunächst ein Vergleich zwischen einzelnen, originalen und normierten Spektren zu sehen. Die Abbildung in 6.1a zeigt das Spektrum der Referenz, wie es von der Kamera aufgenommen wurde. Es ist zu erkennen, dass sich im Wesentlichen die Spektren der Kamera und der Beleuchtung - wie sie in Abbildung 3.2 zu sehen sind - überlagern: Die Peaks der Beleuchtung sind deutlich erkennbar. Ebenso gibt es bei ca. 1400nm einen Tiefpunkt, der durch das Aufnahmespektrum der verwendeten Kamera zu erklären ist. Außerdem ist zu sehen, dass die Intensität in den Randbereichen des Aufnahmebereichs gegen Null geht, was sowohl mit der Sensitivität der Kamera als auch mit der Emission der Beleuchtung in diesen Bereichen zu erklären ist.
Da zur Normierung das Referenzspektrum verwendet wird, ist das normierte Referenz-



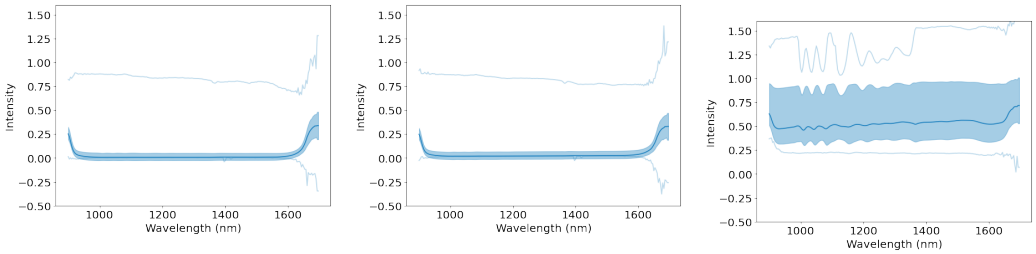
(a) Referenzspektrum

(b) AkzoNobel Aerobase

Abbildung 6.1: Originale (blau) und normierte (orange) Spektren

spektrum wie erwartet nahezu eine Konstante nahe Eins. In beiden Randbereichen

der aufgenommenen Wellenlängen - besonders im Bereich der Wellenlängen größer als $1600nm$ - ist jedoch hohes Rauschen zu sehen. Dies ist mit der geringen Intensität in diesen Bereichen durch fehlende Beleuchtung und mangelnde Sensitivität der Kamera zu erklären.
In Abbildung 6.1 ist außerdem ein beispielhaftes Spektrum des blauen Aerobase-Basecoats von AkzoNobel zu sehen. Das Rauschen im Randbereich tritt ebenso wie bei der Referenz auf. Hier ist die Intensität allerdings über den gesamten Verlauf etwas niedriger als beim Referenzspektrum. Das Spektrum des Aerobase-Lacks besitzt im Vergleich zu allen anderen verwendeten Lacken einen sehr charakteristischen Tiefpunkt bei etwa $1050nm$.
Werden die Spektren mehrerer Pixel derselben Klasse ausgewertet, so sind sich die



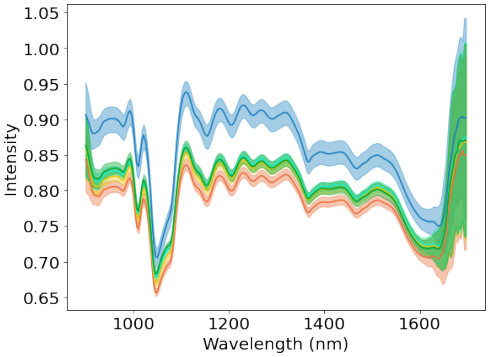
(a) Mankiewicz Alexit H/S Basecoat 411-22 black

(b) AkzoNobel Aerobase RAL 5004

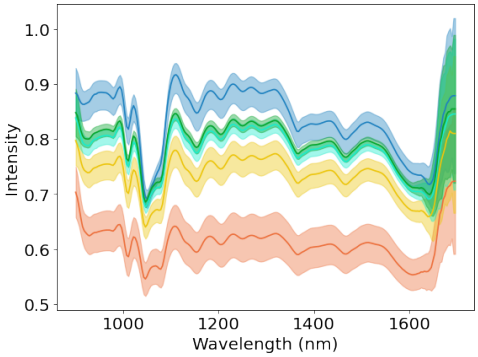
(c) Aluminium

Abbildung 6.2: Durchschnittliche Spektren ausgewählter Klassen (Dunkelblaue Linie: Median; hellblauer Bereich: Standardabweichung; hellblaue Linie: Minimum/Maximum)

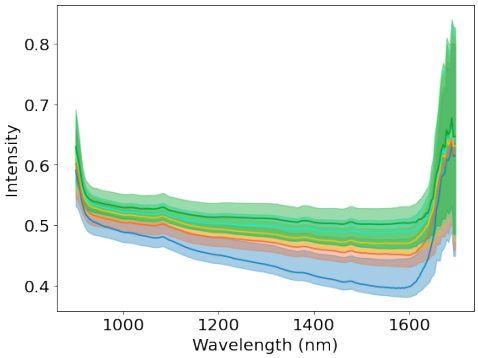
meisten Spektren sehr ähnlich. Es gibt zwar einige starke Abweichungen, die aber auch mit falscher Annotierung oder Übergangsbereichen zu erklären sein könnten, die Standardabweichung ist jedoch bei den meisten Klassen gering.
Bei der Darstellung durchschnittlicher Spektren mit deren Standardabweichung pro Wellenlänge wird wieder das Rauschen in den Randbereichen durch höhere Standardabweichung deutlich.
Während sich die Spektren vieler Lacke durch Intensität, Verlauf oder charakteristische Extrema wie beim Aerobase-Lack auf den ersten Blick einigermaßen gut unterscheiden lassen, fallen beim Vergleich der Spektren auch einige auf, die sich nicht so einfach differenzieren lassen oder andere Probleme mitbringen. Einige Beispiele sind in 6.2 abgebildet. Die Spektren aller Klassen befinden sich im Anhang.
Ein Problem dunkler Lacke wird in den Grafiken 6.2a und 6.2b deutlich: Die Intensität über alle aufgenommenen Wellenlängen ist so gering, dass es kaum Merkmale gibt, um die Lacke zu differenzieren. Dies trifft auf alle schwarzen oder sehr dunklen Lacke sowie darauf aufgetragene Klarlacke zu.
Bei dem Spektrum von Aluminium (siehe Abb. 6.2c) fällt außerdem auf, dass die Standardabweichung über alle Wellenlängen sehr groß ist. Das bedeutet, dass die Aufnahmen von Aluminium sehr unterschiedlich sein können, was bei der Vorhersage zu Verwechslungen mit anderen Oberflächen führen kann.



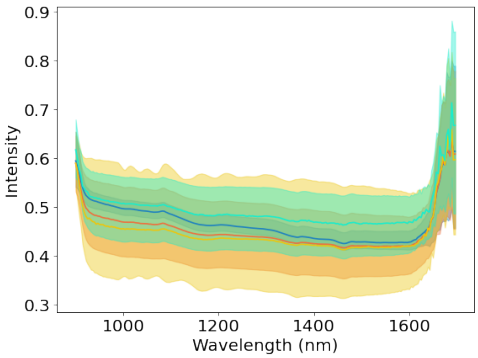
(a) Mankiewicz Alexit H/S Basecoat 411-22 KLM



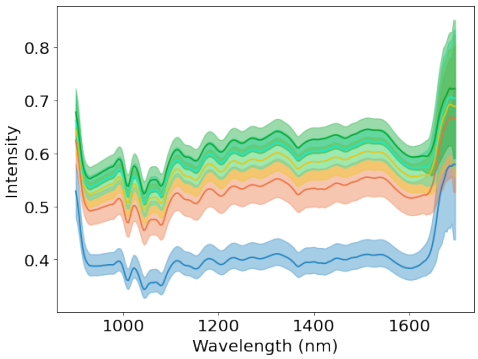
(b) AkzoNobel Aviox CC UVR auf Alexit H/S Basecoat 411-22 KLM



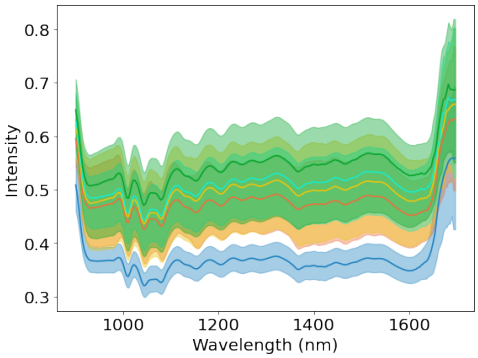
(c) Mankiewicz Alexit H/S Basecoat 411-22 grau



(d) AkzoNobel Aviox CC UVR auf Alexit H/S Basecoat 411-22 grau



(e) AkzoNobel Aerobase



(f) AkzoNobel Aviox CC UVR auf AkzoNobel Aerobase

Abbildung 6.3: Darstellung der Spektren ausgewählter Klassen aufgeteilt nach Laserzyklen (blau: ungelasert; orange: 1; gelb: 2; türkis: 3; grün: 4)

Die bisher dargestellten Spektren stammen alle von ungelaserten Lacken ohne die Verwendung von Klarlack. In Abbildung 6.3 sind sowohl Spektren von unterschiedlichen Laserbearbeitungszyklen als auch Spektren von Basislacken vor und nach dem Auftragen von Klarlack abgebildet.
Zwischen den Spektren, die den Basislack zeigen, und den Spektren, die einen Klarlack über diesem Basislack zeigen, lassen sich nur wenig Unterschiede finden und ohne weitere Analysen kein Muster, wie der Klarlack das Spektrum verändert: Bei einigen führt der Klarlack zu höheren Intensitäten, bei anderen Basislacken wiederum zu niedrigeren. Manchmal hat er eine höhere Streuung zur Folge und manchmal eine leichte Änderung des Verlaufs.

Dafür sind bei einigen Lacken deutliche Unterschiede bei den Laserzyklen zu erkennen. Aber auch hier kommt die Veränderung sehr auf den Lack an: Während sich die Spektren der Laserzyklen beim KLM-blauen Basislack im gesamten Verlauf verändern, wie in Abbildung 6.3a zu sehen, verändert sich das Spektrum beim in Abbildung 6.3c zu sehenden Spektrums des grauen Basislacks von Mankiewicz vor allem im Bereich der höheren Wellenlängen.
Bemerkenswert ist außerdem, dass keine generelle Aussage getroffen werden kann, ob die Laserbehandlung zu einer höheren oder niedrigeren Intensität führt: Während beim KLM-blauen Lack in 6.3b der erste Laserzyklus zu einer deutlich niedrigeren Intensität führt, erhöhen weitere Zyklen die Intensität wieder. Bei den anderen beiden ausgewählten Lacken hingegen führt auch der erste Laserzyklus zu einer Steigerung der Intensität.

6.2 Korrektheit

6.2.1 Übersicht

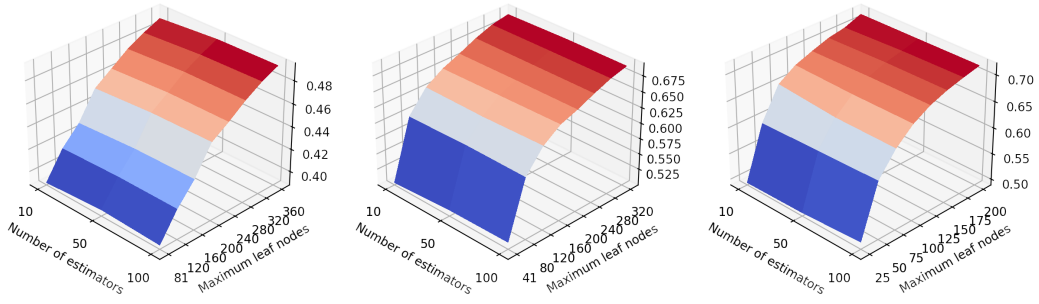
Die trainierten Modelle werden zur Analyse der Korrektheit hierarchisch miteinander verglichen und jeweils das beste ausgewählt. Dazu werden zu Beginn die besten Hyperparameter für die Random Forests mit spektralen Informationen, die Random Forests mit räumlichen und spektralen Informationen, die neuronalen Netze mit spektralen Informationen und die neuronalen Netze mit räumlichen und spektralen Informationen pro Zielklasse auf Basis der Validierungsdaten bestimmt.
Die drei je Modellart ausgewählten Klassifizierer werden im Anschluss miteinander verglichen, um daraus ein Modell pro Modellart auszuwählen. Diese vier Modelle werden zum Schluss wiederum miteinander verglichen.
Bei den Vergleichen geht es jeweils um die Eignung, die Aufgabe zu erfüllen. Dazu werden neben den reinen Metriken pro Datensatz auch Analysen durchgeführt, die die Stärken und Schwächen der Modelle aufdecken, um neben den besten Modellen zusätzlich Erkenntnisse zu gewinnen, welche Modelle sich für welchen Anwendungsfall am besten eigenen.
Für die Analyse wurden viele Plots angefertigt, die für die folgenden Unterkapitel ausgewertet werden. Aus Platzgründen, werden nur die Grafiken dargestellt, die besondere Informationen oder wichtige Unterschiede zwischen den Modellen zeigen. Grafiken, bei denen alle Modelle nahezu die gleichen Ergebnisse erzielt haben, werden weggelassen.

6.2.2 Random Forests mit ausschließlich spektralen Informationen

Vergleich der Modelle pro Zielklasse

Für die Suche nach einem geeigneten Random-Forest-Modell wurden beim Training die Hyperparameter *Anzahl der Bäume* und *Maximale Anzahl der Blätter* variiert. Um eine Aussage über deren Einfluss auf die Korrektheit der Vorhersage zu treffen und ein bestes Modell auszuwählen, werden die Genauigkeit, die Sensitivität, die Spezifität und der F1-Score für jedes trainierte Modell mit Hilfe der Validierungsdaten errechnet.
Zur Auswahl der besten Modelle wird der F1-Score verwendet, da dieser weniger anfällig für über die Klassen ungleich verteilte Beispiele ist als die Genauigkeit.

Die durchschnittlichen F1-Scores für die drei Zielklassen sind in Abbildung 6.4 zu sehen. Es ist erkennbar, dass zwischen der Anzahl der Bäume und der Korrektheit der Modelle mit den ausgewählten Parametern kein Zusammenhang besteht. Die maximale Anzahl der Blätter pro Baum hat hingegen einen großen Einfluss bei allen trainierten Modellen: Je höher die Blattanzahl desto höher wird der F1-Score. Die Verläufe der Genauigkeit, Sensitivität und Spezifität verlaufen ähnlich wie die der F1-Scores. Es ist ebenfalls ein positiver Einfluss der Anzahl der Blätter und kaum Veränderung durch mehr Bäume festzustellen. Für die meisten Modelle gilt, dass die Sensitivität der Modelle etwas höher als die Spezifität ist.
Insbesondere bei der Darstellung der Scores für die Modelle, die zwischen allen Klassen unterscheiden können, sieht der Graph danach aus, als würde eine weitere Erhöhung der Blattzahl zu einer weiteren Verbesserung der Scores führen. Da das jedoch die bereits lange Trainingszeit weiter erhöht, wird an dieser Stelle darauf verzichtet.
Weder der F1-Score noch die anderen berechneten Metriken sagen etwas darüber



(a) Alle Klassen (b) Gelasert/ungelasert (c) Nur Lacke

Abbildung 6.4: F1-Score über die variierten Hyperparameter beim Training der Random Forests für die drei Zielklassen

aus, wie ähnlich die Modelle entscheiden. Es wäre denkbar, dass ein Modell den Hintergrund, der in der späteren Anwendung eher unwichtig sein wird, sehr gut erkennt, dafür aber eine andere Klasse nicht zuverlässig und ein anderes Modell alle Klassen gleich gut klassifizieren kann. Da diese Eigenschaften für die Auswahl jedoch unbedingt berücksichtigt werden sollten, werden die Modelle zusätzlich hinsichtlich ihrer Stärken und Schwächen analysiert. Dazu wird der Mittelwert des F1-Scores über die Klassen von den F1-Scores der einzelnen Klassen subtrahiert und die sich dadurch ergebenden Werte dargestellt. Der Graph für die Modelle, die nur den Lack erkennen, ist in 6.5 abgebildet. Es ist zu erkennen, dass die Modelle die unterschiedlichen Klassen sehr ähnlich gut bzw. schlecht erkennen können. Zwar gibt es große Schwankungen zwischen den Klassen, die Kurven liegen bis auf wenige Ausnahmen aber sehr nah beieinander. Die abweichenden Scores gehören zu Modellen mit 25 oder 50 Blättern, die insgesamt deutlich schlechtere Ergebnisse erreichen. Die Graphen für die beiden anderen Zielklassen, die zusätzlich betrachten, ob die Oberfläche gelasert wurde oder nicht bzw. die Anzahl der Laserzyklen bestimmen, sind ähnlich aufgebaut wie der für die Erkennung der Lacke: Es gibt teils starke Schwankungen, die Verläufe der unterschiedlichen Modelle liegt nah beieinander. Gibt es Abweichungen, so treten diese bei den Modellen mit wenigen Blättern auf. Da diese jedoch durch ihre schlechte Gesamtperformance eher nicht zur Auswahl des besten Modells zur Verfügung stehen, wird diese Abweichung als unkritisch bewertet und die Auswahl des besten Modells

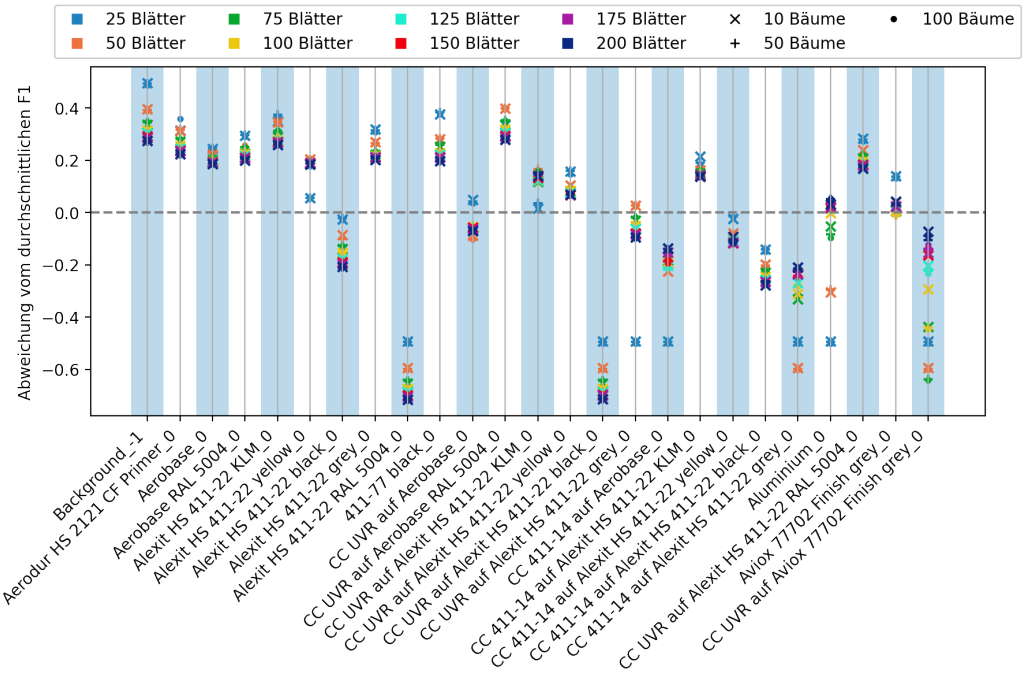


Abbildung 6.5: Abweichung des F1-Scores pro Klasse vom Mittelwert für alle spektralen Random Forest Modelle, die für die Erkennung der Lacke oder Laserzustände trainiert wurden

kann durch die F1-Scores geschehen.

Auswertung der besten Modelle pro Zielklasse

	Anzahl Bäume	Anzahl Blätter	Genauigkeit	F1
Alle Klassen	100	360	0.733	0.493
Gelasert/ungelasert	50	320	0.8154	0.6912
Nur Lacke	100	200	0.8038	0.7171

Tabelle 6.1: Parameter und Scores der besten Random Forest Modelle

Tabelle 6.1 zeigt die Parameter und Scores der Modelle, die unter den Random Forests für die rein spektralen Daten anhand ihres F1-Scores als bestes Modell ausgewählt wurden. Vergleicht man die Modelle untereinander, so ist das Modell zur ausschließlichen Erkennung der Lacke das am besten geeignete. Das Modell zur zusätzlichen Erkennung des Bearbeitungszustands ist ähnlich gut, das zur Differenzierung der Laserzyklen jedoch ist nach den Metriken deutlich schlechter als die beiden anderen. Um ein Modell daraus auszuwählen und abschätzen zu können, ob eine differenziertere Erkennung der Klassen einen schlechteren Score rechtfertigt, sind jedoch weitere Analysen nötig.
Um auch bei diesen Modellen herauszufinden, ob sie ähnliche Stärken und Schwächen oder unterschiedliches Verhalten haben, wird wieder der F1-Score pro Klasse verwendet. Da die Zielklassen der Modelle jedoch alle unterschiedlich sind, lässt sich dieser nicht ohne weiteres vergleichen. Daher wird der Score für die fehlenden Klassen der beiden Modelle mit weniger Klassen geschätzt, indem bei deren Vorhersagen immer diejenige der richtigen spezielleren Klasse angenommen wird, wenn das Modell entsprechend seiner Möglichkeiten die richtige Klasse ausgewählt hat. Die so errechneten

Scores sind für die drei ausgewählten Modelle in Abbildung 6.6 zu sehen.
Die Grafik zeigt, dass die Modelle dieses Mal nicht alle Klassen etwa gleich gut

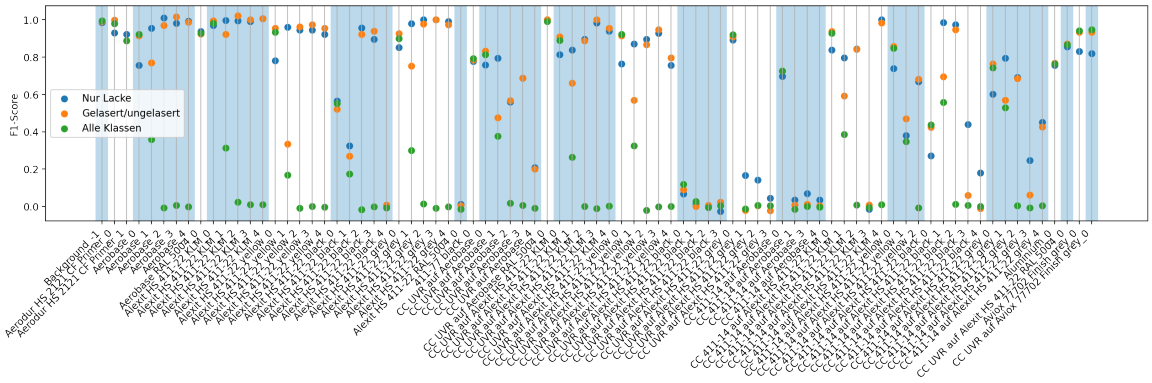


Abbildung 6.6: F1-Scores der besten Random Forest-Modelle über alle Klassen

erkennen können. Stattdessen gibt es teils starke Unterschiede zwischen den Modellen pro Klasse. Das Modell zur Erkennung der Lacke und das Modell, das zusätzlich erkennt, ob die Oberfläche mit dem Laser bearbeitet wurde, weisen ähnliche Scores auf. Das Modell zur Bestimmung des Laserzyklus weicht hingegen stärker ab.
Auffällig ist, dass das Modell zur Unterscheidung aller Klassen zwar meist die ungelaserten Oberflächen ähnlich gut erkennt wie die beiden anderen Modelle, bei den gelaserten Oberflächen jedoch deutlich schlechtere Scores hat. Dieser Vergleich ist jedoch mit Vorsicht zu betrachten, da die Vorhersage des Modells zur Erkennung alle Klassen bereits als falsch klassifiziert gilt, wenn die Klasse sich nur um einen Laserzyklus unterscheidet. Bei den anderen Modellen hingegen wird der Score für die Laserzyklen lediglich geschätzt und bei ihnen ist es nicht relevant, ob sie z.B. statt des zweiten Laserzyklus den dritten vorhersagen.
Bei der Betrachtung der Scores des Modells zur Bewertung, ob ein gelaserter oder ungelaserter Zustand vorliegt, fällt auf, dass dem Modell die Differenzierung zwischen dem ungelaserten und dem einmal gelaserten Lack schwer fällt, während es die mehrfach gelaserten Oberflächen wieder zuverlässiger erkennen kann. Dieses könnte an nur kleinen Abweichungen zwischen dem ungelaserten und dem einmal gelaserten Lack liegen, die mit steigender Zyklenanzahl stärker werden.
Bis auf das Modell zur Erkennung aller Klassen können die Modelle die meisten Klassen ohne Klarlack gut unterscheiden. Schwächen sind bei den dunklen Lacken wie schwarz und dem mitternachtsblauen RAL-Farbton 5004 zu erkennen, die zum Teil von allen Modellen sehr unzuverlässig erkannt werden. Auch bei der Erkennung des Klarlacks gibt es viele Schwächen. Die Vorhersage funktioniert unzuverlässiger als bei den farbigen Lacken und es gibt einige Kombinationen, die nicht erkannt werden. Es lässt sich allerdings nicht verallgemeinernd sagen, dass auf einem Basislack Klarlacke generell nicht erkannt werden. Während zum Beispiel der UVR-Klarlack auf dem schwarzen Alexit-Basislack von allen Modellen fast nicht erkannt wird, sind die Scores bei dem anderen verwendeten Klarlack deutlich besser.

Mit den bisherigen Auswertungen konnte analysiert werden, welche Klassen von den Modellen besser und welche schlechter erkannt werden und ob es dabei Unterschiede zwischen den Modellen gibt. Da die Klassen bei der vorliegenden Anwendung jedoch

auch semantische Zusammenhänge haben, sollten zusätzlich bei Verwechslungen untersucht werden, ob eine komplett andere Klasse vorhergesagt wurde oder zum Beispiel nur der Laserzyklus um einen abweicht, da einige Fehler schwerere Auswirkungen haben können, während andere kaum Bedeutung haben.
Diese Analysen können mit der Konfusionsmatrix getätigt werden. Da die Konfusionsmatrizen jedoch aufgrund der vielen Klassen sehr groß werden und der Vergleich der Matrizen verschiedener Modelle dementsprechend komplex ist, werden an dieser Stelle stattdessen die Informationen zu Fehlerklassen zusammengefasst. Diese Fehlerklassen orientieren sich an den semantischen Informationen der Klasse und einigen häufigen und naheliegenden Verwechslungen von Klassen. Die Fehler werden in sechs Klassen aufgeteilt: Testdaten, bei denen Lack und eventueller Klarlack korrekt vorhergesagt werden und nur die Anzahl der Laserzyklen falsch eingeschätzt wird, werden in der Kategorie *falsche Laserzyklen* gezählt. Daneben gibt es Kategorien für den *richtigen Klarlack auf falschem Basislack* und das Gegenteil *falscher Klarlack auf richtigem Basislack*. Außerdem gibt es Kategorien für Beispiele, bei denen ein Klarlack erkannt wurde, obwohl keiner vorhanden ist (*Klarlack erfunden*) oder bei denen andersherum nur der Basislack erkannt wurde, obwohl darauf eigentlich ein Klarlack aufgetragen worden ist (*Klarlack übersehen*). Alle übrigen Fehler werden in der letzten Kategorie gesammelt (*Andere*). Bei den Kategorien zwei bis fünf wird die Anzahl der Laserzyklen ignoriert.
Die Anzahlen der Fehler pro Klasse und Modell wird in Abbildung 6.7 dargestellt.

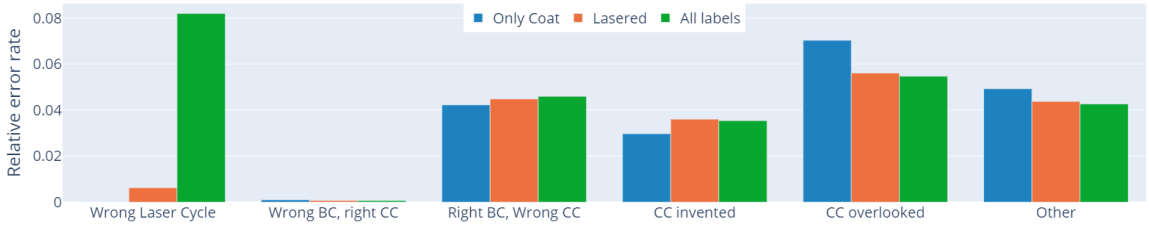


Abbildung 6.7: Anzahl der falsch klassifizierten Spektren pro Fehlerklasse und Modell zur Analyse der Verwechslungen der Random Forest Modelle

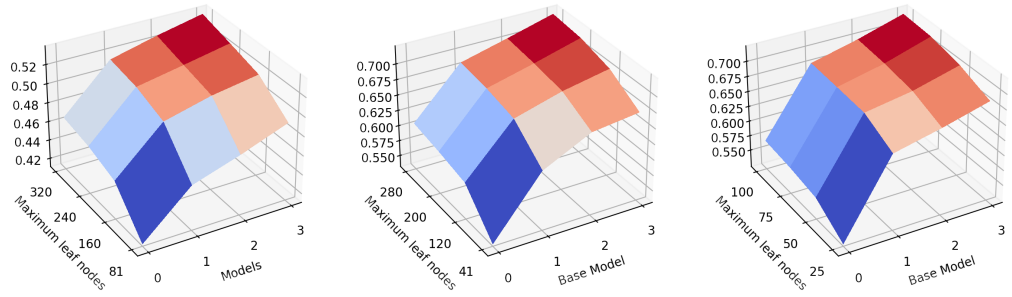
Es gibt fast keine Testdaten, bei denen der richtige Klarlack auf falschem Basislack klassifiziert wurde. Das spricht dafür, dass die Modelle nicht den Klarlack auf unterschiedlichen Basislacken verallgemeinert erkennen, sondern das spezifische Spektrum eines Klarlacks auf einem bestimmten Basislack lernen. Anders herum gibt es einige Testdaten, bei denen der richtige Basislack ausgewählt, darauf aber der falsche Klarlack erkannt wurde. Dabei nimmt die Sicherheit der Erkennung mit steigender Klassenanzahl leicht ab. Alle Modelle sehen bei einigen Spektren einen Klarlack, wo keiner ist oder übersehen ihn. Während das Modell zur ausschließlichen Differenzierung selten nicht vorhandenen Klarlack erkennt, übersieht es ihn am häufigsten. Das Modell zur Differenzierung aller Klassen schneidet hier im Durchschnitt am besten ab. Bei der Fehlklassifizierung des Laserzyklus ist der Unterschied zwischen den Modellen besonders groß. Während das Modell zur Unterscheidung der Lacke diese Fehler nicht machen kann und das Modell zum Bestimmen, ob der Lack per Laser behandelt wurde, wenige Fehler macht, ist das Modell zur Unterscheidung aller Klassen hier sehr unsicher. Bei den restlichen Fehlern schneidet das Modell zur

Differenzierung aller Klassen am besten ab. Also sollte - selbst, wenn die Erkennung der Laserzyklen nicht relevant für die Anwendung ist, trotz des schlechten F1-Scores das Modell zur Unterscheidung der Laserzyklen gewählt werden, da es im Vergleich mit den beiden anderen Modellen weniger andere Fehler macht.

6.2.3 Random Forests mit spektralen und räumlichen Informationen

Vergleich der Modelle pro Zielklasse

Die Basismodelle der Random Forests für die Verarbeitung spektraler und räumlicher Informationen wurden auf Basis der Analysen der Random Forests mit ausschließlich spektralen Informationen ausgewählt. Da diese gezeigt haben, dass die Anzahl der Bäume kaum Einfluss auf die Modelle hat, wurden nur Modelle mit 10 Bäumen pro Random Forest ausgewählt. Diese wurden gleichmäßig auf die variierte Anzahl der Blätter verteilt, sodass aus den gewählten Anzahlen der Blätter aus 6.2.2 der erste, vierte, sechste und achte Parameter ausgewählt wurde. Diese Modelle sind die Modelle $0 - 3$. Die Anzahl der Bäume für das neu zu trainierende Modell wurde auf 10 festgelegt. Die Anzahl der Blätter wird wieder variiert.
Die F1-Scores der so trainierten Modelle sind in 6.8 dargestellt. Es gibt wieder einen



(a) Alle Klassen (b) Gelasert/ungelasert (c) Nur Lacke

Abbildung 6.8: F1-Score über die variierten Basismodelle und Anzahl der Blätter beim Training der Random Forests mit räumlichen und spektralen Informationen für die drei Zielklassen

Zusammenhang des F1-Scores mit der Anzahl der Blätter des Modells: Sowohl eine Erhöhung der Anzahl der Blätter im Basismodell als auch mehr Blätter im Random Forest der zweiten Stufe führen zu einer Verbesserung der Ergebnisse.
Um sicher zu gehen, dass es keine großen Unterschiede in den Stärken und Schwächen der einzelnen Modelle gibt, wird wieder die Abweichung des F1-Scores pro Klasse vom Mittelwert des jeweiligen Modells betrachtet. Es gilt für alle Zielklassen, dass es einige Ausreißer nach unten gibt, die aber nur bei den Modellen auftreten, die ein Basismodell mit wenigen Blättern verwenden.

Auswertung der besten Modelle pro Zielklasse

Von den Random Forests, die sowohl räumliche als auch spektrale Daten verarbeiten können, wurden pro Zielklasse jeweils die mit den meisten Blättern im Basismodell und im Modell der zweiten Stufe ausgewählt. Die Modelle sind in Tabelle 6.9 zusammengefasst.

	Anzahl Blätter Basismodell	Anzahl Blätter	Genauigkeit	F1
Alle Klassen	360	320	0.7607	0.5365
Gelasert/ungelasert	320	280	0.843	0.7245
Nur Lacke	200	100	0.8248	0.7212

Abbildung 6.9: Parameter und Scores der besten Random Forest Modelle mit räumlichen und spektralen Daten

Von den Metriken her ist das Modell zur Unterscheidung der Lacke in gelasert und ungelasert das beste, während das Modell zur Unterscheidung der Laserzyklen am schlechtesten abschneidet. Auch hier soll die Eignung der Modelle für die unterschiedlichen Klassen abgeschätzt werden, indem die F1-Scores für die unbekannten Klassen geschätzt werden. Das Ergebnis ist in Abbildung 6.10 dargestellt.
Es ist wieder zu erkennen, dass das Modell die Lacke ohne Klarlacke eher sicher

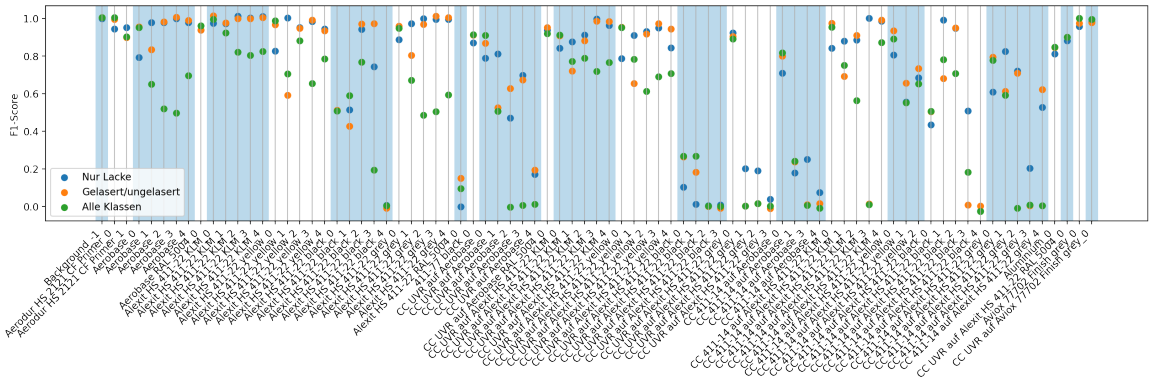


Abbildung 6.10: F1-Scores der besten Random Forest-Modelle mit räumlichen und spektralen Daten über alle Klassen

erkennen kann. Die Erkennung der Klarlacke fällt den Modellen etwas schwerer. Schwierigkeiten gibt es außerdem bei den sehr dunklen Lacken. Zusätzlich gilt wieder, dass das Modell zur Erkennung, ob die Oberfläche mit dem Laser behandelt wurde, mit dem ersten Laserzyklus Probleme hat, während es die unbearbeitete Oberfläche und die weiteren Laserzyklen besser erkennt. Das Modell zur Differenzierung aller Oberflächen erkennt in den meisten Fällen den unbearbeiteten Zustand etwas besser als die gelaserten Oberflächen.
Ob es daran liegt, dass es lediglich bei den Laserzyklen falsche Einschätzungen abgibt, wird mit der Betrachtung der Fehleranzahlen pro Fehlerkategorie in Abbildung 6.11 überprüft.
Hier ist wieder zu erkennen, dass das Modell zur Erkennung aller Klassen die größten Probleme mit der Erkennung des korrekten Laserzyklus hat. Dieses Mal hat jedoch das Modell zur Differenzierung der Oberflächen in gelasert und ungelasert in fast allen Kategorien die wenigsten Fehler. Da dieses Modell zudem die höchsten Scores erreicht und das Modell zur Erkennung aller Klassen Schwierigkeiten bei der korrekten Klassifizierung des Laserzyklus hat, sollte es ausgewählt werden, wenn die Bestimmung des Laserzyklus nicht zwingend notwendig ist.

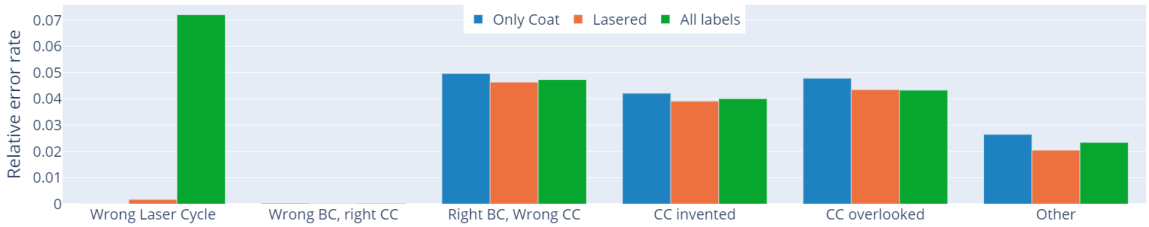
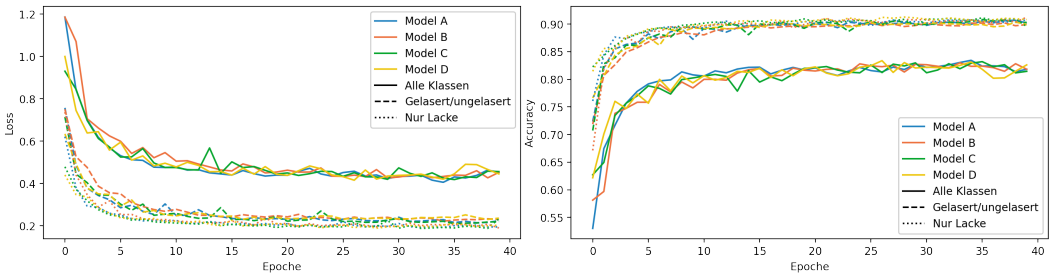


Abbildung 6.11: Anzahl der falsch klassifizierten Spektren pro Fehlerklasse und Modell zur Analyse der Verwechslungen der Random Forest Modelle mit räumlichen und spektralen Daten

6.2.4 Neuronale Netze mit ausschließlich spektralen Informationen

Vergleich der Modelle pro Zielklasse



(a) Validation Loss (b) Validation Accuracy

Tabelle 6.2: Loss- bzw. Accuracy-Plot der Trainings der mit einzelnen Spektren trainierten neuronalen Netze

Beim Training neuronaler Netze geben die Plots von Loss und weiteren Metriken bereits wertvolle Informationen darüber, wie geeignet das Modell zum Erfüllen der Aufgabe ist. In Abbildung 6.2 sind die Losses und die Genauigkeit der Modelle über die Epochen mit dem Validierungsdatensatz aufgetragen.
Alle Trainings haben einen gleichmäßigen Verlauf und nähern sich im Verlauf der trainierten Epochen einem Maximalwert an. Die Modelle zur Erkennung aller Klassen liefern dabei deutlich geringere Genauigkeiten als die anderen Modelle, während die Modelle zur Erkennung der Lacke nur leicht besser abschneiden als die zur Differenzierung zwischen bearbeiteten und unbearbeiteten Oberflächen.
Die Sensitivität, Spezifität und der F1-Score der Modelle je einer Zielklasse liegen sehr nah beieinander. Um sicherzustellen, dass die Modelle sich nicht durch Stärken in einzelnen Klassen voneinander unterscheiden, wird die Abweichung des F1-Scores pro Klasse überprüft. Bei den Modellen zur Erkennung der Lacke sind hier keine signifikanten Abweichungen festzustellen. Bei den neuronalen Netzen, die zusätzlich erkennen, ob eine Oberfläche mit dem Laser bearbeitet wurde, gibt es bei einigen Klassen größere Unterschiede. Zwar ist Modell B nach Abbildung 6.12 bei den meisten Klassen gegenüber den anderen Modellen leicht überlegen. Bei mehreren Klassen, insbesondere bei dem Alexit-Basislack im RAL-Farbton 5004 ist das Modell B deutlich schlechter als die beiden anderen. Wegen der Schwächen bei einigen Modellen und annähernd gleichen durchschnittlichen F1-Scores sollte Modell B eher nicht für

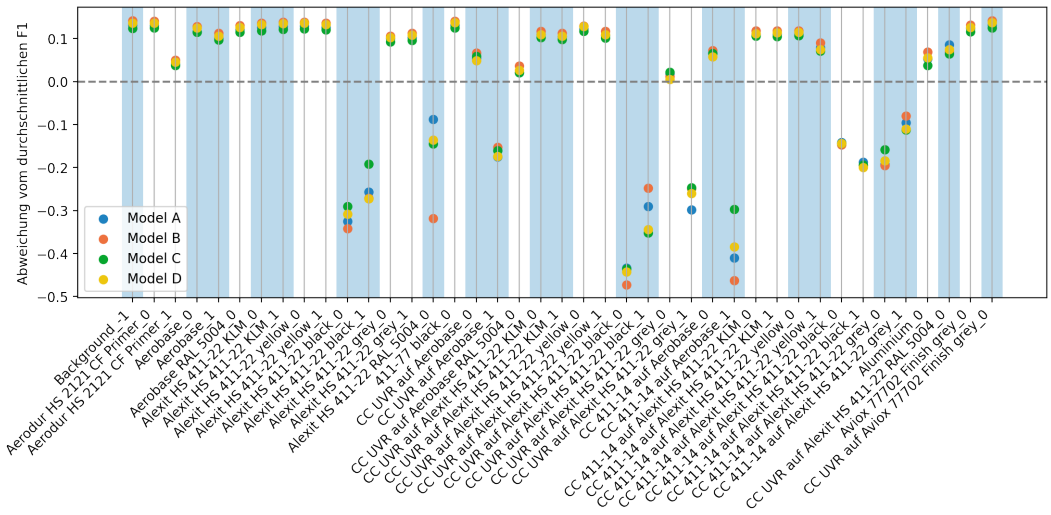


Abbildung 6.12: Abweichung des F1-Scores pro Klasse vom Mittelwert für alle spektralen neuronalen Netze, die für die Erkennung der Lacke oder Laserzustände trainiert wurden

die Zielklasse mit Unterscheidung in gelasert und ungelasert ausgewählt werden. Bei den neuronalen Netzen zur Erkennung aller Klassen sind zwar auch einige Unterschiede zwischen den Modellen zu sehen. Jedoch lässt sich kein Modell ausmachen, das in vielen Fällen deutlich besser oder schlechter als der Rest ist. Daher werden die besten Modelle für die Erkennung der Lacke und die Erkennung aus allen Modellen nach dem F1-Score ausgewählt und das beste Modell zur Differenzierung in gelaserte und ungelaserte Oberflächen ohne Modell B.

Auswertung der besten Modelle pro Zielklasse

Bei den neuronalen Netzen, die einzelne Spektren auswerten, wurde durch die vorherigen Bewertungen für jede Zielklasse eine andere Modellarchitektur ausgewählt. Durch die nah beieinander liegenden Metriken hat das jedoch zunächst wenig Aussagekraft über die Eignung der unterschiedlichen Architekturen. Das Modell zur Unterscheidung der Lacke und das Modell zur Unterscheidung der Oberfläche in gelasert und ungelasert haben ähnliche Genauigkeiten. Das Modell zur Erkennung aller Klassen ist jedoch nach den Scores deutlich ungenauer, wie schon in den Loss-Plots zu erkennen war.
Ob das Modell zur Differenzierung aller Klassen deswegen weniger gut geeignet für

	Bestes Modell	Genauigkeit	F1
Alle Klassen	C	0.8344	0.6855
Gelasert/ungelasert	C	0.9082	0.871
Nur Lacke	D	0.9127	0.8989

Tabelle 6.3: Parameter und Scores der besten neuronalen Netze spektralen Daten

die Aufgabe ist, wird mit weiteren Analysen überprüft. Dazu werden zunächst wie in den vorherigen Modellen die F1-Scores der einzelnen Klassen (siehe Abbildung 6.13) miteinander verglichen. Auch hier ist schnell ersichtlich, dass die beiden Modelle

mit weniger Klassen viele Lacke zuverlässig voneinander unterscheiden können und sich in vielen Fällen ähnlich sind. Das Modell zur Unterscheidung in gelaserte und ungelaserte Lacke kann den ersten Laserzyklus etwas schlechter erkennen, ebenso wie das Modell zur Erkennung aller Klassen ungelaserte Oberflächen oft etwas besser erkennen kann als gelaserte Lacke. Ein Vergleich dieser Werte unter den Modellen ist jedoch aus den bereits erläuterten Problemen mit Vorsicht zu betrachten.
Darum werden die Fehler erneut in Klassen eingeteilt und die Anzahl der falsch

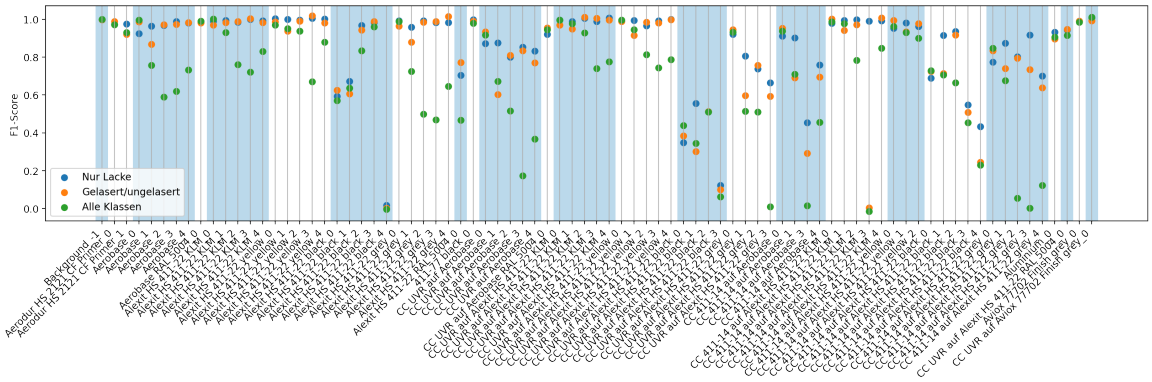


Abbildung 6.13: F1-Scores der besten neuronalen Netze mit spektralen Daten über alle Klassen

klassifizierten Daten in diesen Fehlerklassen miteinander verglichen. Welche Fehler die Modelle jeweils gemacht haben, ist in Abbildung 6.14 zu sehen. Keines der Modelle erkennt häufig den falschen Basislack unter dem richtigen Klarlack. Die Anzahl der Vorhersagen, bei denen die Modelle komplett falsch liegen, ist außerdem ebenfalls niedrig. Dabei gilt, dass die Anzahl der Fehler mit der Anzahl der vorherzusagenden Klassen ansteigt. Die Testdaten, bei denen das Modell mit seiner Einschätzung der Laserzyklen falsch lag, ist potentiell durch die größere Auswahlmöglichkeit bei dem Modell zur Erkennung aller Klassen deutlich am höchsten. Bei den Fehlerklassen, bei denen die Modelle den falschen Klarlack erkennen, einen Klarlack übersehen oder ihn sehen, obwohl er nicht da ist, lässt sich kein eindeutiger Favorit unter den Modellen ausmachen. Für die Auswahl des am besten geeigneten Modells sollte hier der Anwendungsfall genau definiert und dadurch die Fehlerklassen priorisiert werden. Im Durchschnitt ist das Modell zur Differenzierung in gelasert und ungelasert am besten.

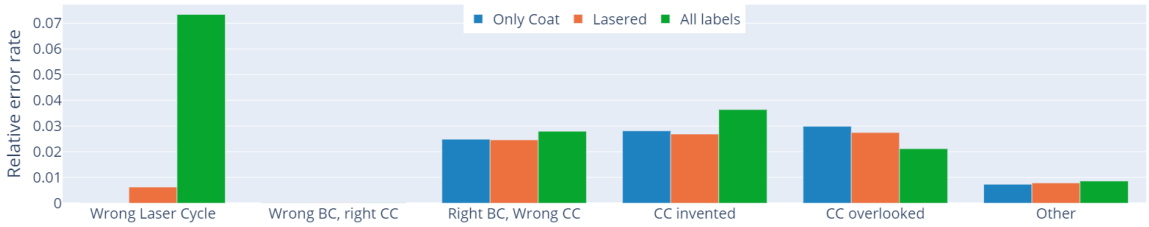
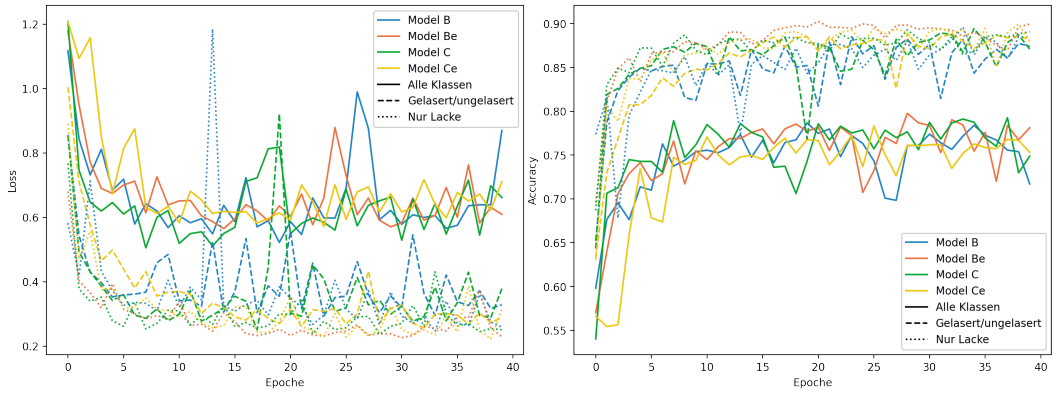


Abbildung 6.14: Anzahl der falsch klassifizierten Spektren pro Fehlerklasse und Modell zur Analyse der Verwechslungen der neuronalen Netze mit spektralen Daten

6.2.5 Neuronale Netze mit spektralen und räumlichen Informationen

Vergleich der Modelle pro Zielklasse



(a) Validation Loss (b) Validation Accuracy

Abbildung 6.15: Loss- bzw. Accuracy-Plot der Trainings der spektralen mit räumlichen Daten trainierten neuronalen Netze

In den Diagrammen des Loss und der Genauigkeit in Abbildung 6.15 ist zu sehen, dass diese Werte sich zwar während des Trainings verbessern, aber dennoch stark schwanken. Während die Modelle zur Erkennung aller Klassen deutlich schlechter sind als die der beiden anderen Zielklassen, ähneln sich die Metriken dieser Modelle. Wie auch in Abbildung 6.16 ist das Modell zur ausschließlichen Erkennung der Lacke leicht besser als das zur zusätzlichen Erkennung des Bearbeitungszustands der Oberfläche.
Wie bereits in den vorherigen Kapiteln werden die Modelle pro Zielklasse unterein-

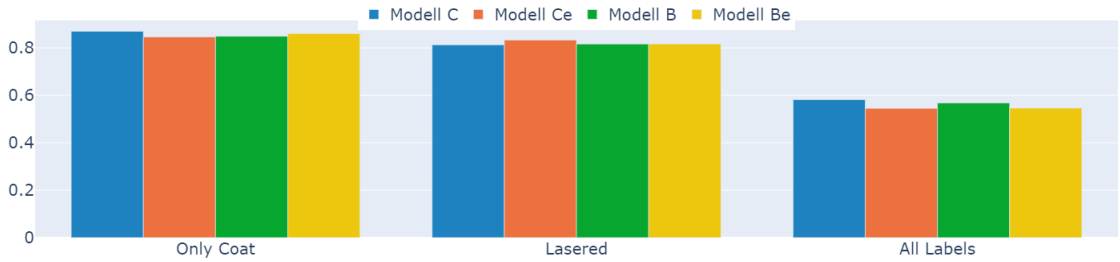


Abbildung 6.16: F1-Score über neuronale Netze, die mit räumlichen und spektralen Informationen trainiert wurden

ander anhand der Abweichung des F1-Scores vom durchschnittlichen F1-Score und ihrem durchschnittlichen F1-Score verglichen. Bei den Modellen zur ausschließlichen Differenzierung der Lacke erreicht Modell B die höchsten Scores. In Abbildung 6.17 fällt außerdem auf, dass dieses Modell den Alexit-Basislack im RAL-Farbton 5004 deutlich besser erkennen kann als die übrigen Modelle. Auch wenn das Modell bei einigen anderen Lacken etwas schlechter abschneidet, heben diese beiden Umstände das Modell B etwas hervor.

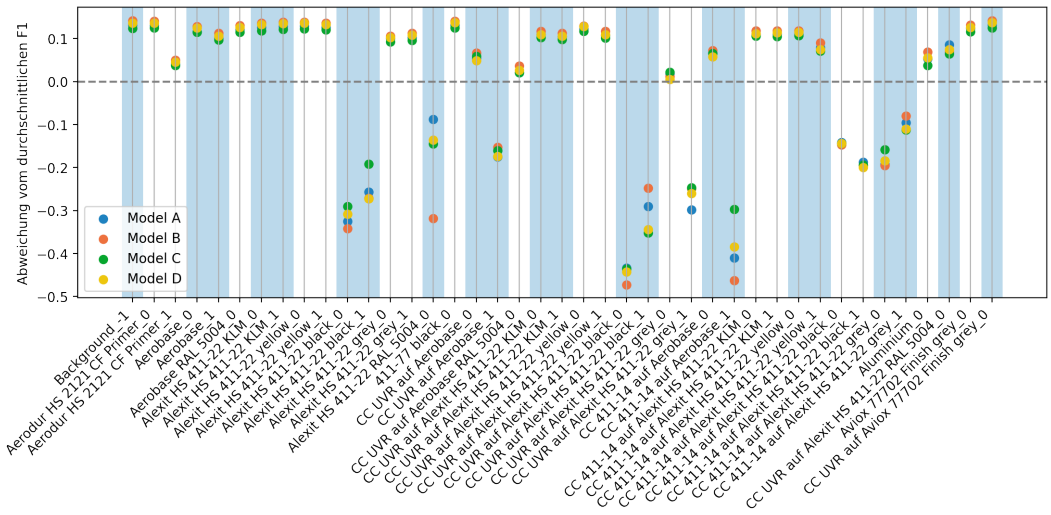


Abbildung 6.17: Abweichung des F1-Scores pro Klasse vom Mittelwert für alle neuronalen Netze, die für die Erkennung der Lacke mit spektralen und räumlichen Daten trainiert wurden

Bei den Modellen der anderen zwei Zielklassen gibt es zwar Unterschiede zwischen den F1-Scores einiger Klassen, jedes Modell hat jedoch seine Schwächen, sodass sich kein eindeutiger Favorit auswählen lässt. Aus diesem Grund werden die Modelle durch den höchsten durchschnittlichen F1-Score ausgewählt.

Auswertung der besten Modelle pro Zielklasse

	Bestes Modell	Genauigkeit	F1
Alle Klassen	B	0.7867	0.5748
Gelasert/ungelasert	Be	0.8932	0.8331
Nur Lacke	B	0.8961	0.8699

Tabelle 6.4: Parameter und Scores der besten neuronalen Netze mit spektralen und räumlichen Daten

Als beste Modelle wurden Modelle mit der Architektur B ausgewählt. Darunter sind zwei Modelle, die erst spät räumlich falten und ein Modell, das im ersten Schritt die räumlichen Informationen zusammenfasst. Da die verwendeten Scores jedoch nahe beieinander lagen, hat dieser Umstand nur begrenzte Aussagekraft über die Eignung der anderen Architekturen.
Auch bei den neuronalen Netzen, die mehrere benachbarte Spektren zur Auswertung eines Pixels erhalten, gilt, dass die Genauigkeit der Modelle der Zielklassen *Nur Lacke* und *Gelasert/ungelasert* ähnlich sind und das ausgewählte Modell zur Erkennung aller Klassen schlechter abschneidet.
Die detaillierten Scores für die einzelnen Klassen sind in Abbildung 6.18 zu sehen. Auch in dieser Grafik sind die - wahrscheinlich mit den Zielen der Modelle zusammenhängenden Probleme - zu sehen, dass das Modell zur Erkennung aller Klassen mit den Laserzyklen $1 - 4$ Schwierigkeiten hat und das Modell zur Differenzierung in gelasert und ungelasert schlechtere Werte im ersten Laserzyklus erreicht. Es gibt jedoch auch unabhängig davon größere Unterschiede in den Scores der einzelnen Modelle, die oft bei den dunklen Lacken auftreten, bei denen der F1-Score insgesamt

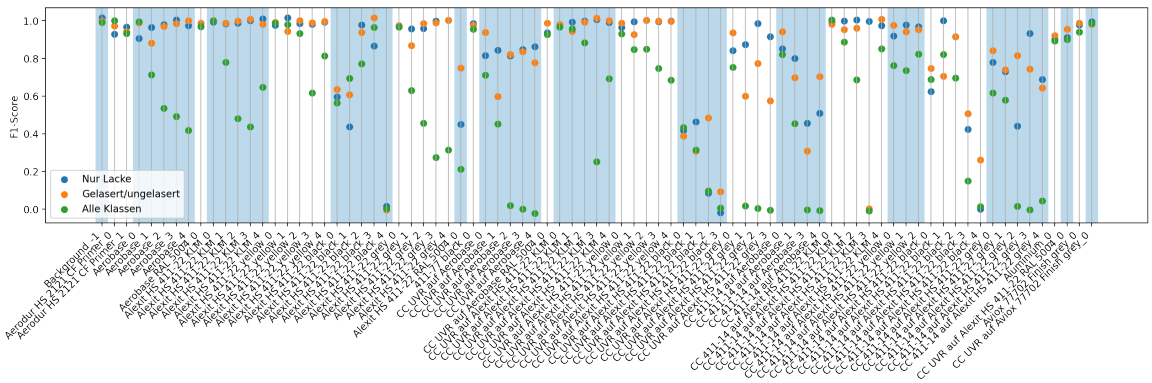


Abbildung 6.18: F1-Scores der besten neuronalen Netze mit spektralen und räumlichen Daten über alle Klassen

niedriger ist. Es lässt sich jedoch auf dieser Basis kein eindeutiger Favorit als bestes Modell auswählen.
Dazu werden im nächsten Schritt die Verwechslungen bei der Vorhersage anhand der

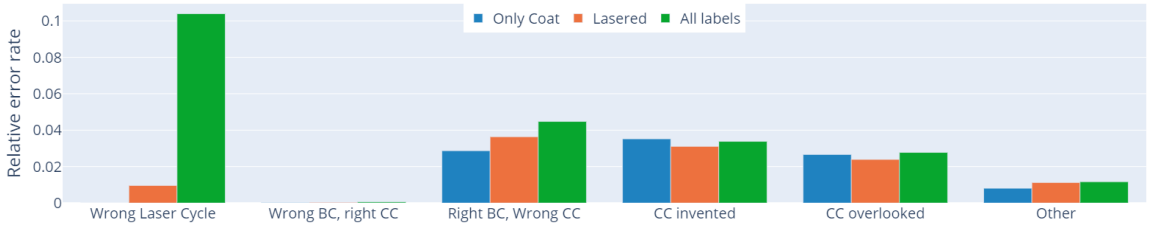


Abbildung 6.19: Anzahl der falsch klassifizierten Spektren pro Fehlerklasse und Modell zur Analyse der Verwechslungen der neuronalen Netze mit spektralen und räumlichen Daten

Fehlerkategorien, die in 6.19 abgebildet sind, analysiert. Während alle Modelle einige Fehler rund um die Vorhersage des Klarlacks machen, treten kaum Fehler auf, bei denen die Vorhersage den falschen Basislack erkennt. Insbesondere das Modell zur Erkennung der Lacke macht hier wenige Fehler. Sollte für den Laser die Erkennung des Klarlacks irrelevant sein, so sollte dieses Modell gewählt werden. In den anderen Fällen macht das Modell zur Differenzierung in gelasert und ungelasert die wenigsten Fehler, bzw. bildet den Mittelwert zwischen den beiden anderen Modellen. Aus diesem Grund wird dieses Modell hier zunächst bevorzugt.

6.2.6 Vergleich der besten Modelle

In den vorherigen Kapiteln wurden die besten Modelle je Modellart anhand der Scores, ihrer individuellen Stärken und Schwächen und bei der Vorhersage gemachten Fehler ausgewählt. Diese sollen nun miteinander verglichen werden.
Bis auf ein Modell gehören alle Modelle zu der Zielklasse *Gelasert/ungelasert*. Bei den Random Forests wurden jeweils die komplexesten Modelle ausgewählt, während es bei den neuronalen Netzen noch komplexere Modelle zur Auswahl gegeben hätte. Das nach den Scores beste Modell ist das ausgewählte neuronale Netz mit Modellarchitektur C, das einzelne Spektren auswertet. Während bei den Random Forests die

	Zielklasse	Modell(-parameter)	Accuracy	F1
Random Forest (spektral)	Alle Klassen	100 Bäume, 360 Blätter	0.733	0.493
Random Forest (spektral und räumlich)	Gelasert/ ungelasert	320 Blätter im Basismodell, 280 Blätter in zweiter Stufe, 10 Bäume	0.843	0.7245
Neuronales Netz (spektral)	Gelasert/ ungelasert	Modell C	0.9082	0.8989
Neuronales Netz (spektral und räumlich)	Gelasert/ ungelasert	Modell Be	0.8932	0.8331

Tabelle 6.5: Nach Korrektheit ausgewählte beste Modelle je Modellart

Auswertung von benachbarten Spektren die Genauigkeit deutlich verbessern konnte, hat sie bei den neuronalen Netzen zu einer leicht verminderten Leistung geführt.
Beim Vergleich der F1-Scores der besten Modelle in Abbildung 6.20 sind wie nach

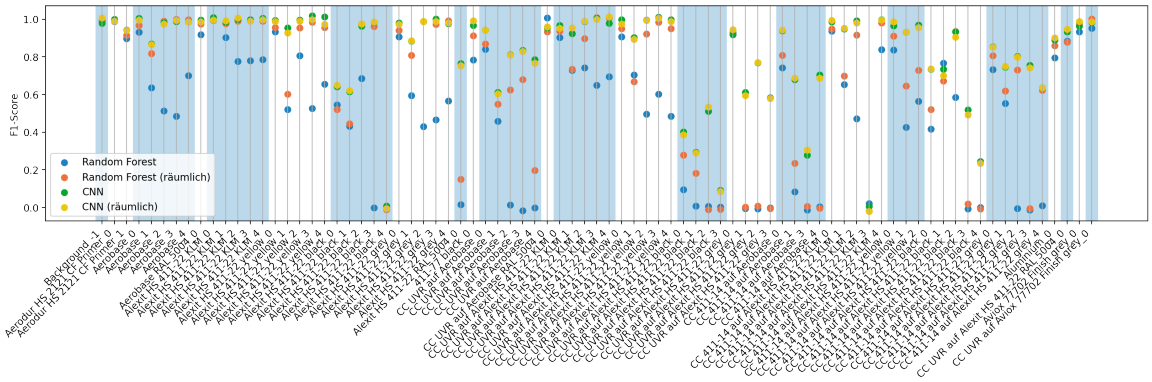


Abbildung 6.20: Vergleich der F1-Scores pro Klasse der besten Modelle

den Scores erwartet die neuronalen Netze besser als die Random Forest-Modelle. Die schlechteren Scores des Random Forest mit spektralen Daten bei den gelaserten Klassen sind allerdings hauptsächlich darauf zurückzuführen, dass dieses Modell im Gegensatz zu den anderen mit allen Klassen trainiert wurde und dementsprechend mehr Verwechslungsgefahr besteht. Das führt auch dazu, dass die Metriken des Modells deutlich niedriger sind. Trotzdem zeigen beide Random Forest-Modelle insbesondere bei den aufgetragenen Klarlacken deutlich schlechtere Ergebnisse als die neuronalen Netze und erkennen die Klassen teilweise gar nicht, wohingegen die neuronalen Netze eine relativ hohe Trefferquote haben.
Alle Modelle haben bei vielen Klarlacken Erkennungsprobleme und bis auf beim schwarzen 411-77-Topcoat Schwierigkeiten bei den dunklen Lacken.
In welchen Fehlertypen sich diese Verwechslungen äußern, ist in Abbildung 6.21 zu sehen. Bei der korrekten Erkennung der Laserzyklen hat es das erste Random Forest Modell deutlich schwerer als die anderen, da es mit allen Klassen trainiert wurde und dadurch mehr Auswahlmöglichkeiten hat. Bei den übrigen also potentiell kritischsten Fehlern, bei denen das Modell den Basislack vertauscht hat, liegt das neuronale Netz mit nur spektralen Informationen knapp vor dem mit zusätzlichen räumlichen Informationen. Auch bei den anderen Kategorien macht es eher wenige Fehler, sodass

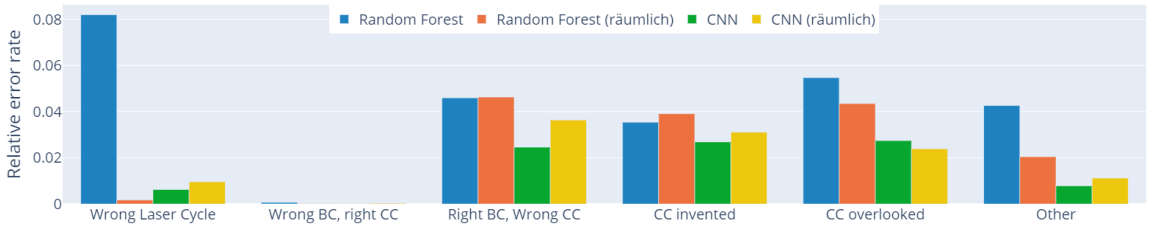


Abbildung 6.21: Anzahl der falsch klassifizierten Spektren pro Fehlerklasse für die ausgewählten Modelle

es ohne weitere Priorisierung der Fehlerkategorien auf Basis der Korrektheit als bestes ausgewählt wird.
In Abbildung 6.22 ist dessen Konfusionsmatrix zu sehen. Da das Netz trotz der einzelnen Betrachtung der Spektren mit Hypercubes trainiert wurde, um das Training zu beschleunigen, enthält der Testdatensatz einige Spektren mit undefinierter Klasse, die während der Annotierung übersehen wurden oder nicht zuzuordnen waren. Um nicht die in den Hypercubes weiteren annotierten Spektren ausschließen zu müssen, wurden diese auch zur Auswertung verwendet. In den vorherigen Analysen wurden die einzelnen nicht annotierten Spektren jedoch nicht verwendet, um die Metriken zu berechnen. Da die nicht annotierten Bereiche sowie die dort vorliegenden Lacke jedoch bekannt sind und nur die Zuordnung von Pixel und Klasse unbekannt ist, wird *undefined* in der Konfusionsmatrix nicht ausgeschlossen.
Wie durch die hohen Scores bereits zu erwarten war, sieht auch die Konfusionsmatrix nach einer guten Vorhersage aus: Die meisten Felder auf der Diagonalen sind dunkelblau, was einer Erkennungsrate von ca. 100% entspricht. Die eher helleren Felder liegen bei dem schwarzen Alexit-Basislack, dem Alexit-Basislack in RAL-Farbton 5004, dem gelaserten UVR-Klarlack auf Aerobase, dem UVR-Klarlack auf schwarzem Alexit-Basislack und auf gelasertem grauen Basislack, dem gelaserten 411-14 Klarlack auf Aerobase oder dem Alexit-KLM und dem 411-14 Klarlack auf schwarzem oder grauen Basislack. Die meisten dieser Auffälligkeiten lassen sich in zwei Gruppen aufgeteilen: zum einen in dunkle Lacke und zum anderen in gelaserte Klarlacke. Schaut man sich näher an, mit welchen Lacken die letztere Gruppe verwechselt wird, so kann man feststellen, dass das in vielen Fällen die gelaserten Basislacke unter den Klarlacken sind.
Auch in den anderen Fällen dreht sich die Verwechslung oft um den Klarlack: Bei den Basislacken sagt das Modell einen nicht vorhandenen Klarlack voraus und bei vorhandenem Klarlack hat es den falschen Klarlack oder den darunter liegenden Basislack vorhergesagt. Eine Verwechslung der Basislacke ist ohne Zahlenwerte in der Konfusionsmatrix nicht zu erkennen und tritt dementsprechend kaum auf.
Bei der Bewertung der nicht annotierten Beispiele passt die Zuordnung bis auf die Vorhersage des Aluminiums zu den bekannten nicht annotierten Bereichen. Es ist außerdem zu sehen, dass nur zu der Undefined-Zeile, nicht aber zu der Undefined-Spalte, Werte zugeordnet wurden. Das bedeutet, dass das Modell erfolgreich gelernt hat, dass Undefined zwar in der Annotation vorkommen kann, es jedoch nicht für Daten vorhergesagt werden soll.

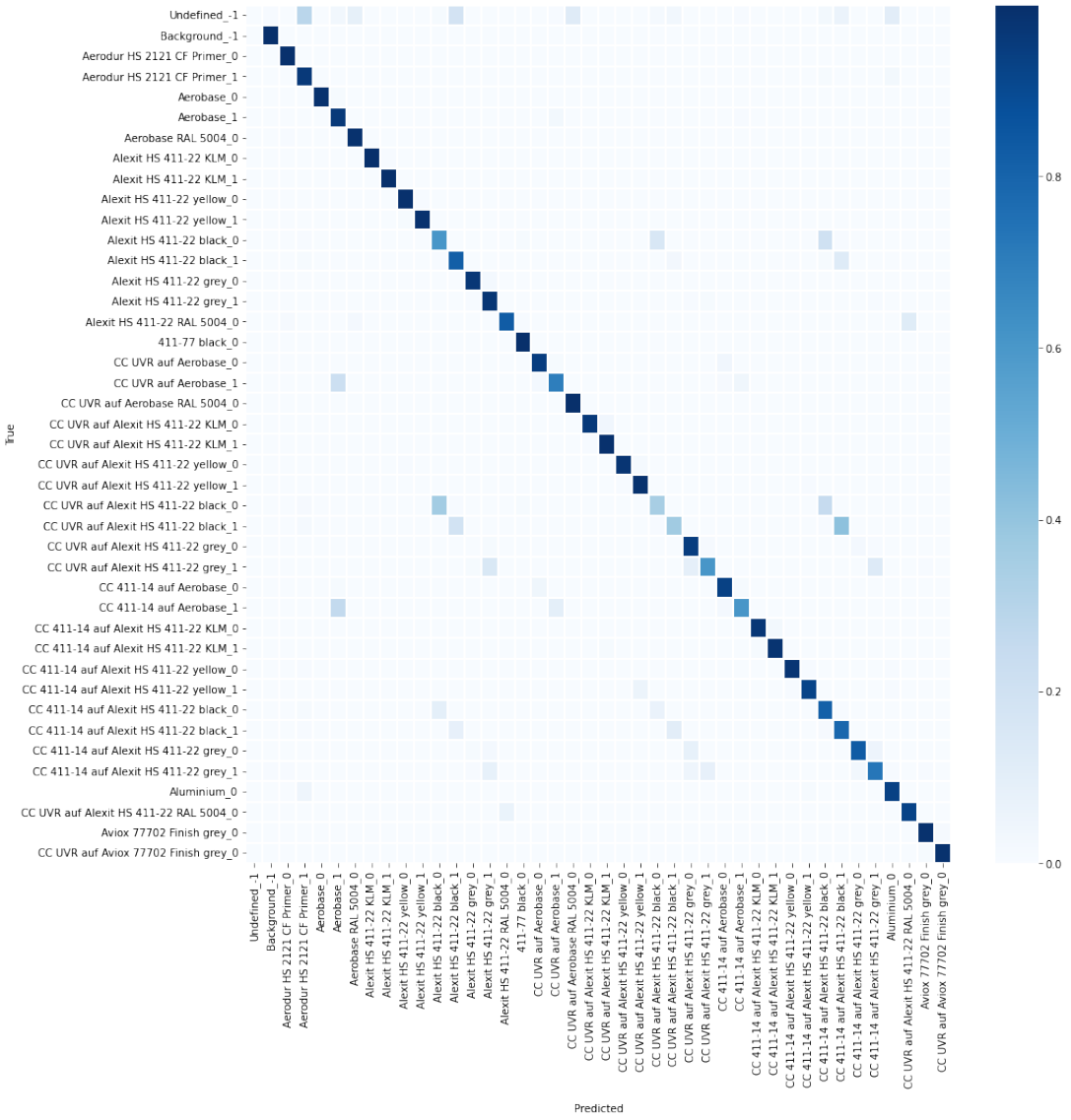


Abbildung 6.22: Konfusionsmatrix des neuronalen Netzes zur Vorhersage der Lacke und Einschätzung der Oberfläche in gelasert und ungelasert auf Basis einzelner Spektren

6.2.7 Optische Betrachtung der Vorhersagen von ganzen Proben

Vorhersagen des Random Forests

In diesem Kapitel sollen einige Klassifikationen des besten Random Forests zur Differenzierung in gelaserte und ungelaserte Oberflächen zusätzlich optisch überprüft werden, um zu erkennen, ob die Vorhersage auch abseits der Metriken plausible Ergebnisse liefert und Hinweise zu finden, worin Fehlerkennungen begründet liegen. Die Eindrücke der optischen Betrachtung stützen die errechneten Fehlerkategorien: Große Teile der Proben werden richtig eingeordnet. Wenn es Fehlklassifikationen gibt, so betreffen sie häufig den Klarlack und fast nie den Basislack.
Ein Beispiel für eine vorhergesagte Probe ist in Abbildung 6.23 zu sehen. Das Modell kann den Bereich ohne Klarlack auf der linken Seite von den beim Lasern übrig gebliebenen Streifen mit Klarlack großenteils gut differenzieren. Die gelaserten Bereiche werden erkannt und auch die Fetzen, die vom Klarlack übrig geblieben sind, erkennt

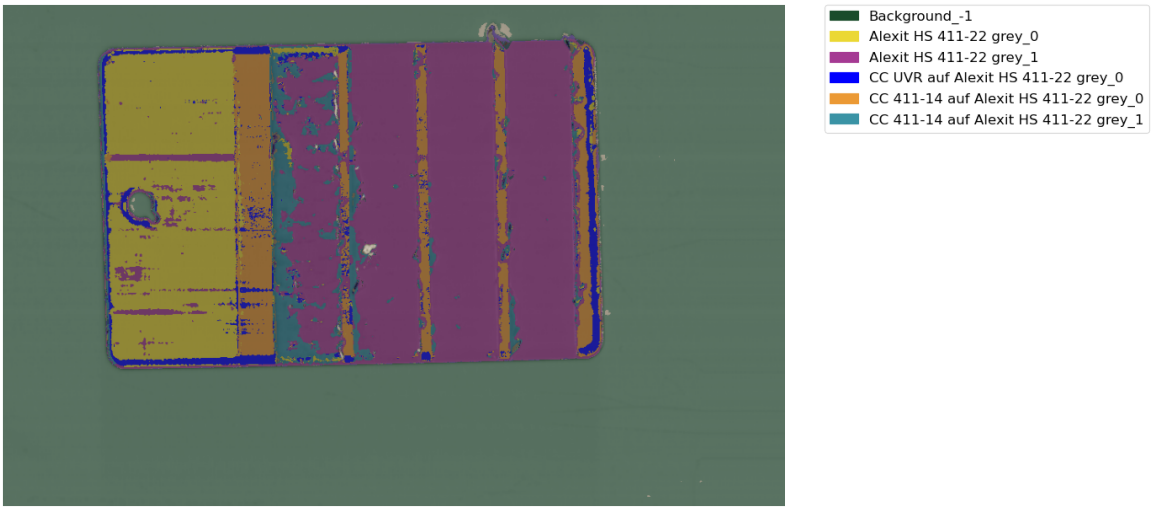


Abbildung 6.23: Vorhersage eines Probenblechs mit Mankiewicz 411-14 Klarlack auf grauem Basislack durch einen Random Forest

das Modell. Rund um den abstehenden Lackfetzen am oberen Rand der Probe wird jedoch Aluminium statt Hintergrund erkannt.
Es ist auch zu erkennen, dass das Modell in einigen Pixeln den falschen Klarlack erkannt hat. Diese Art von Fehler sowie das Erkennen von nicht vorhandenem Klarlack oder das Übersehen von vorhandenem Klarlack ist bei der Klassifizierung von mehreren Proben zu erkennen und entspricht den in 6.21 erkannten Fehlerkategorien. Besondere Schwierigkeiten hat das Modell mit den Rändern der Probe. Hier ist das Blech stark abgerundet, was die Reflektion des Infrarotlichts beeinflussen und damit die Abweichungen erklären könnte. Weitere Fehlklassifikationen gibt es teilweise beim Übergang zweier Lacke.
In Abbildung 6.24 ist die Klassifizierung einer Kachel aus dem Testset der in 6.23

(a) Vorhersage (b) Annotation

Abbildung 6.24: Detaillierter Vergleich zwischen Vorhersage durch einen Random Forest und Annotation einer Kachel

abgebildeten Probe zu sehen. Die Kachel befindet sich mittig über dem ungelaserten Streifen zwischen den Streifen, die ein- bzw. zweimal gelasert wurden. Der weiße Streifen auf der linken Hälfte des Bildes ist eine Irritation, die durch einen hervorstehenden Klarlackfetzen entstanden ist.
Die größte Abweichung, die zwischen Klassifizierung und Annotation auffällt, ist dass in der Annotation der rechte Teil als zweimal gelasert angegeben wird und in der Vorhersage nicht. Dies liegt daran, dass das gewählte Modell nur zwischen gelasert

und ungelasert unterscheiden kann und daher gelaserte Oberflächen korrekterweise als einmal gelasert klassifiziert.
Ein weiterer größerer Unterschied ist die Klassifizierung des ungelaserten mittleren Streifens im Vergleich zu dessen Annotation. Insbesondere im unteren Bereich hat das Modell hier den Klarlack übersehen und stattdessen einmal gelaserten Basislack darin erkannt.
Desweiteren sind kleinere Abweichungen zwischen Annotation und Vorhersage zu sehen. Bei dem Klarlackrest in der oberen rechten Ecke wurde zum Beispiel einige Pixel mehr als Klarlack annotiert, als klassifiziert. Jedoch sieht es im durch die Annotation durchscheinenden Hyperspektralbild so aus, als gehörten tatsächlich etwas weniger Pixel zu dem Klarlack. Dementsprechend hätte das Modell in diesem Fall die Daten besser klassifiziert als die Annotation.
Außerdem werden um den reflektierenden Lackfetzen herum mehrere Pixel zu dem

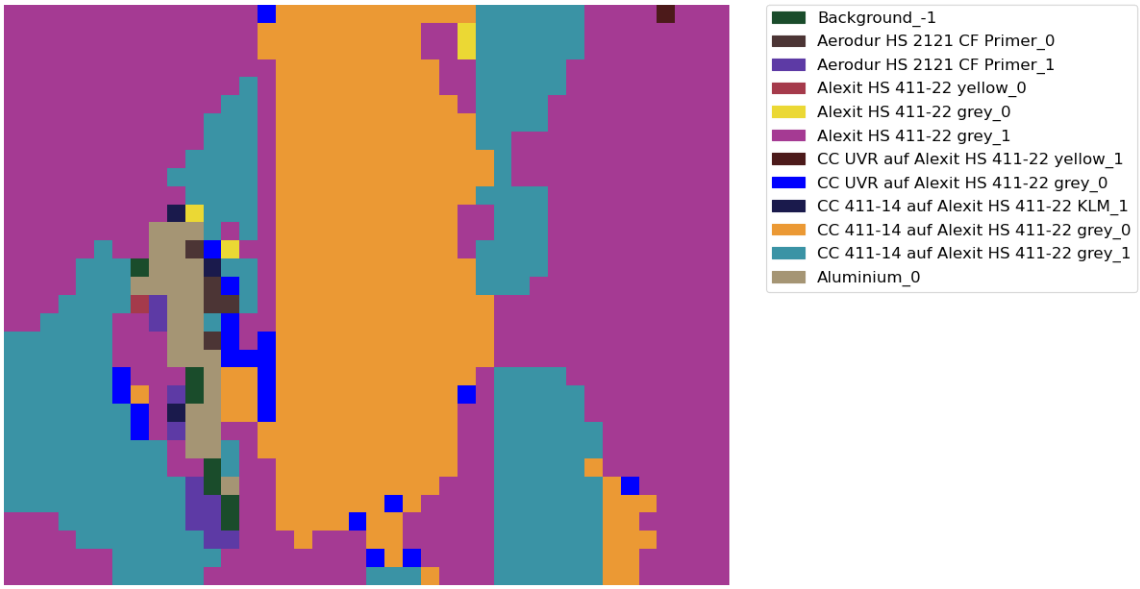


Abbildung 6.25: Detailansicht der durch den Random Forest vorhergesagten Kachel

auf der Probe nicht vorhandenen Aviox UVR-Klarlack zugeordnet. Wird die Klassifizierung nicht als halb transparentes Overlay über dem Hyperspektralbild dargestellt, sondern als Bild ohne Transparenz wie in Abbildung 6.28 so lässt sich erkennen, dass das Modell in diesem Bereich größere Probleme hatte, als bei dem Rest des Bildes. In diesem Bereich werden sehr viele verschiedene Oberflächen vor allem aber Aluminium erkannt, was sich durch die starke Reflektion des Aluminiums mit einem breiten Spektrum erklären lassen könnte.

Vorhersagen des Neuronalen Netzes

Das gleiche Bild mit denselben Ausschnitten wird durch das neuronale Netz mit der Modellarchitektur C zur Differenzierung in gelaserte und ungelaserte Oberflächen klassifiziert.
Wie in Abbildung 6.26 zu sehen, ist auch dieses Modell in der Lage, den Großteil der Oberflächen korrekt zu klassifizieren. Den Scores entsprechend ist die Einordnung durch das neuronale Netz noch etwas besser als durch den Random Forest. Der ungelaserte Bereich ohne Klarlack wird deutlich besser erkannt und auch mit den

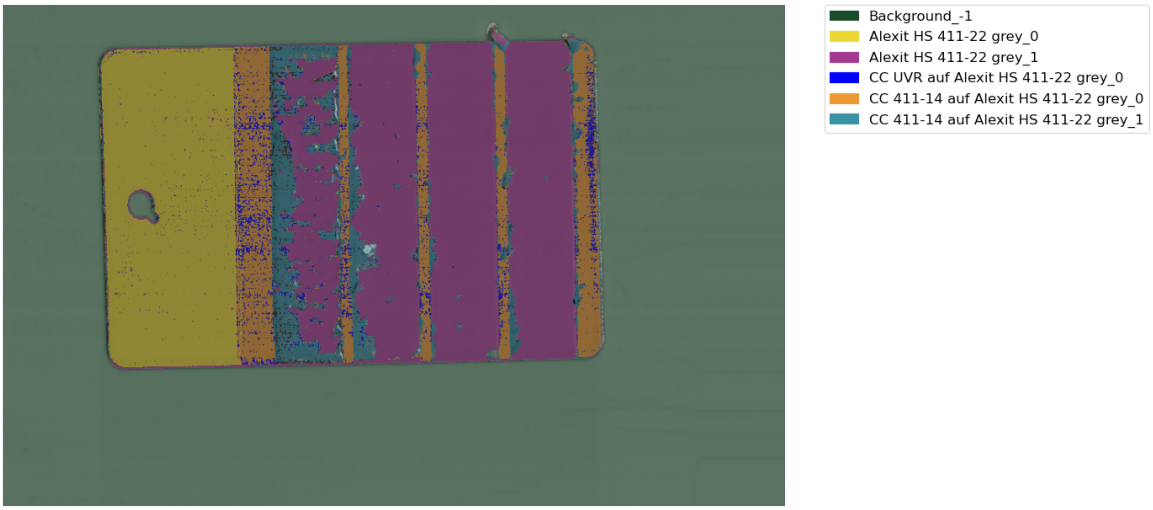


Abbildung 6.26: Vorhersage eines Probenblechs mit Mankiewicz 411-14 Klarlack auf grauem Basislack durch ein neuronales Netz

Rändern der Probe hat das Modell deutlich weniger Probleme. Trotzdem erkennt auch das neuronale Netz bei Übergängen zweier Lacke teilweise falsche Klassen.
Ein ähnliches Bild ergibt sich bei der detaillierten Betrachtung einer Kachel wie in Abbildung 6.27a. Es gibt nur kleine Abweichungen zwischen Klassifikation und Annotation und auch hier sieht es so aus, als würde das Modell teilweise korrektere Ergebnisse liefern als die Annotation.
Das neuronale Netz hat ebenfalls mit dem stark reflektierenden Lackfetzen Probleme

(a) Vorhersage (b) Annotation

Abbildung 6.27: Detaillierter Vergleich zwischen Vorhersage durch ein neuronales Netz und Annotation einer Kachel

(siehe Abbildung 6.28) und bewertet die Oberfläche mit ähnlichen Klassen.

6.3 Geschwindigkeit

6.3.1 Übersicht

Die Geschwindigkeitsanalysen haben das Ziel, bewerten zu können, welche Modelle den Anforderungen genügen. Da der Roboter jedoch nicht gleichzeitig lasern und hyperspektrale Bilder aufnehmen kann, ist noch nicht ganz klar, wie die Prozesse gestaltet werden sollen. Eine Möglichkeit ist, dass der Laser der Kamera nachgeführt

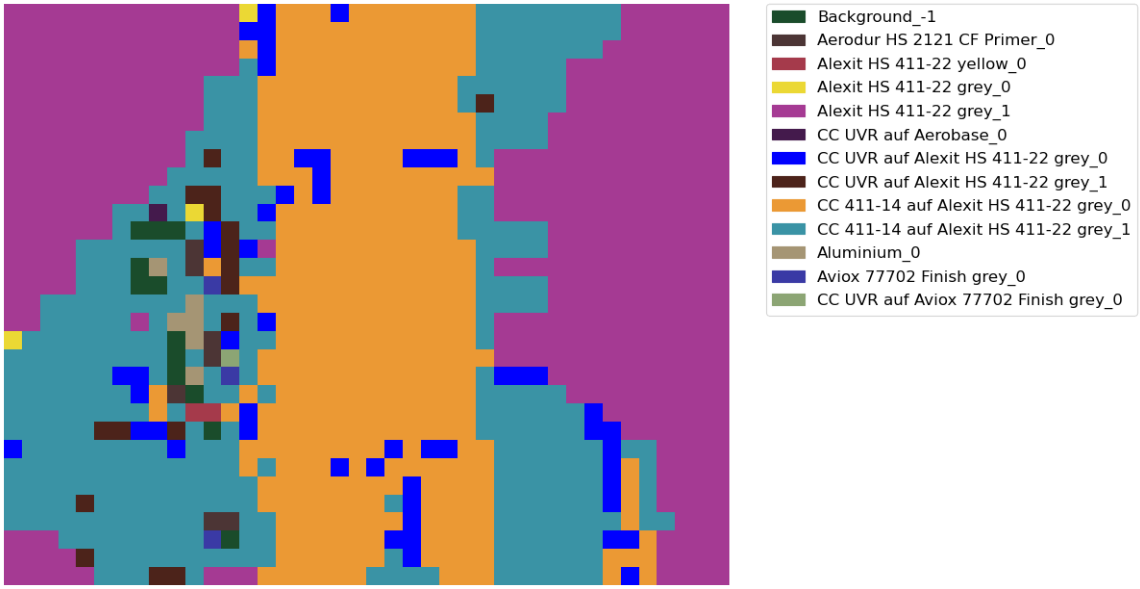


Abbildung 6.28: Detailansicht der vorhergesagten Kachel

wird. Das bedeutet, dass die Modelle nur wenige Zeilen parallel verarbeiten können. Es wäre aber auch möglich, dass der Roboter nach dem Scannen der Oberfläche automatisch sein Werkzeug wechselt, um im Anschluss die Laserbearbeitung durchführen zu können. In diesem Fall könnten mehrere Zeilen parallel verarbeitet werden. Daher wurden die Analysen sowohl mit ganzen Bildern, die 1000 Zeilen mit 640 Spektren umfassen, als auch mit jeweils drei zusammenhängenden Zeilen durchgeführt.

6.3.2 Random Forests mit ausschließlich spektralen Informationen

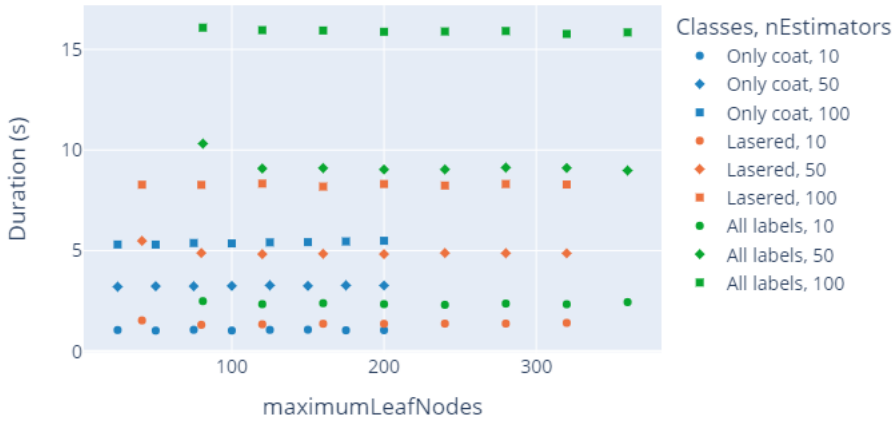


Abbildung 6.29: Dauer der Bewertung eines Bildes (1000px*640px) mit den Random Forests

Bei der Analyse der Bewertungszeiten in Abbildung 6.29 fallen zwei Abhängigkeiten auf. Während die Anzahl der Blätter pro Baum scheinbar keinen Einfluss auf die Dauer hat, wird sie von der Anzahl der Klassen und der Anzahl der Bäume beeinflusst. Aufgrund weniger Anzahlen lässt sich der Zusammenhang nicht eindeutig bestimmen,

aber in beiden Fällen scheint die Dauer linear mit den jeweiligen Anzahlen zu steigen. Bei der Anzahl der Bäume passt das zu der Aussage über die Komplexität der Vorhersage in Kapitel 4.2.
Da die andere Größe, von der die Komplexität der Vorhersage abhängt, die maximale Tiefe der Bäume ist, liegt der Verdacht nahe, dass die Erhöhung der Anzahl der Klassen zu tieferen Bäumen führt. Für den jeweils ersten Baum der Random Forests

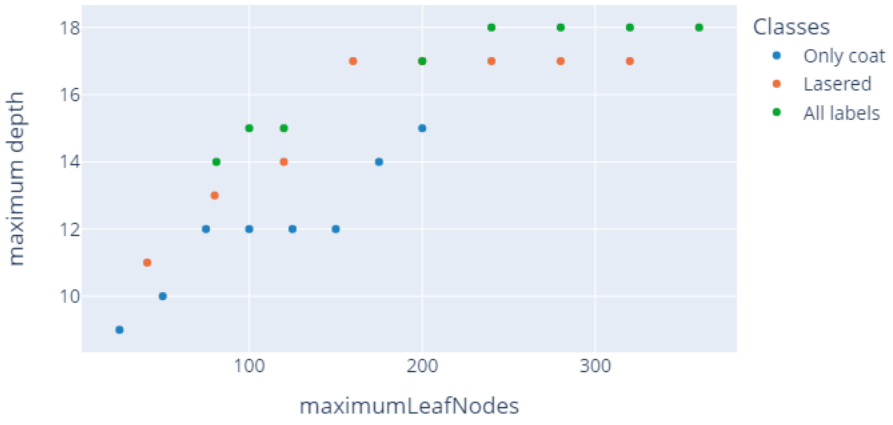


Abbildung 6.30: Tiefe der Bäume in Abhängigkeit der Klassenanzahl und Blattknoten

mit zehn Bäumen ist die Tiefe in Abbildung 6.30 dargestellt. Sie bestätigt diese Vermutung: Die Modelle, die mehr Klassen differenzieren können, sind bei gleicher oder sehr ähnlicher Blattanzahl tiefer als die Modelle, die nur wenige Klassen unterscheiden können.
Die in Abbildung 6.29 dargestellten Zeiten beziehen sich darauf, dass die Vorhersage durch den Random Forest einmal für ein Bild der Größe 1000px*640px aufgerufen wird. Bei dem Aufruf mit lediglich drei Zeilen a 640px wird die Vorhersage deutlich langsamer. Der Faktor, mit dem das geschieht, hängt weder von der Anzahl der Bäume noch von der Anzahl der Blätter, jedoch von der Anzahl der Klassen ab: Bei den Modellen, die lediglich die Lacke unterscheiden, dauert die Vorhersage so etwa elfmal so lang, um alle 1000 Zeilen zu bewerten. Die Modelle zur Unterscheidung in gelasert und ungelasert benötigen $8 - 9$ mal so lang und die Modelle zur Differenzierung aller Klassen etwa 4 mal so lang, um das ganze Bild in Abschnitten mit jeweils drei Zeilen zu bewerten. Dieses könnte mit dem Overhead durch wiederholte Methodenaufrufe erklärt werden. Dass sich die Faktoren mit dem Anstieg der Anzahl der Klassen verringern, könnte mit der proportional zur Komplexität steigenden Dauer pro Bewertung zusammenhängen.

6.3.3 Random Forests mit spektralen und räumlichen Informationen

Bei der Auswertung der Dauer der Bewertung durch die Random Forests, die zusätzlich die benachbarten Spektren zur Bewertung eines Pixels erhalten, wie in Abbildung 6.31 zu sehen, zeigt sich ein ähnliches Bild wie in Kapitel 6.3.2: Da hier die Anzahl der Bäume nicht variiert wurde, fällt dieses als Einflussgröße weg. Jedoch zeigt sich

Abbildung 6.31: Dauer der Bewertung eines Bildes (1000px*640px) mit den Random Forests, die räumliche und spektrale Daten verarbeiten

wieder ein Anstieg der Dauer bei der Erhöhung der Anzahl der Klassen. Auch hier erklärt Abbildung 6.32 diesen Anstieg damit, dass die Bäume zur Vorhersage von mehr Klassen eher eine tiefere Struktur haben.
Während die vergleichbaren Random Forests mit zehn Bäumen aus 6.29 je nach

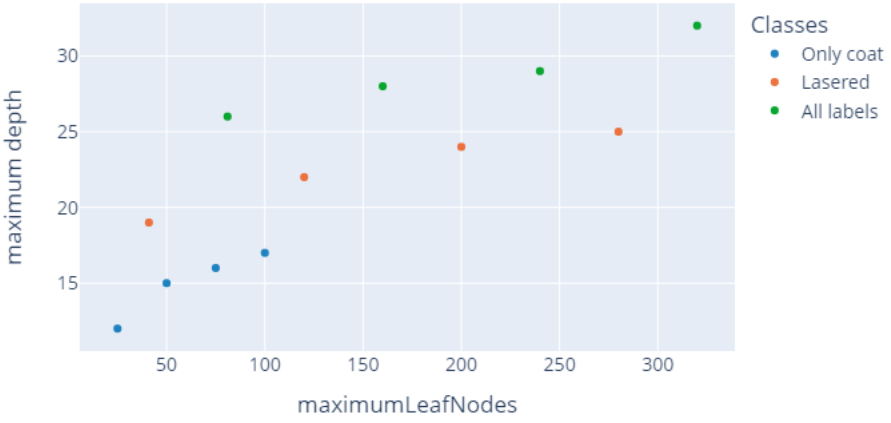


Abbildung 6.32: Tiefe der Bäume in Abhängigkeit der Klassenanzahl und Blattknoten bei den Random Forests, die räumliche und spektrale Daten verarbeiten

Anzahl der Klassen $1 - 2.5s$ pro Bild gebraucht haben, benötigen die Modelle hier etwa $3.5 - 7.5s$, um ein Bild mit 1000 Zeilen zu bewerten.
Auch bei diesen Modellen geht es deutlich schneller, wenn das Modell einmal viele Zeilen bewerten soll, als wenn es mehrfach wenige Zeilen erhält. Die Faktoren, um die sich die Dauer erhöht, sind mit 12 für alle Klassen, 20 für gelasert/ungelasert und knapp 27 für die ausschließliche Differenzierung zwischen den Lacken noch deutlich höher als bei den Modellen, die nur einzelne Spektren betrachten.

6.3.4 Neuronale Netze

Abbildung 6.33: Dauer der Bewertung eines Bildes (1000px*640px) mit den trainierten neuronalen Netzen

Bei den neuronalen Netzen konnten teilweise nicht die gesamten Bilder in einer Vorhersage bewertet werden, da der GPU-Speicher dafür nicht ausreichte. Daher wurden bei den rein spektralen Modellen B, C und D halbe und bei den spektralen und räumlichen Modellen zehntel Bilder berechnet und die Gesamtdauer hochgerechnet. Die gemessenen Zeiten, die gebraucht werden, um das Bild zu bewerten, unterscheiden sich stark, weswegen die y-Achse in Abbildung 6.33 logarithmisch ist. Es zeigt sich, dass Modell A, welches nicht in der Lage ist, mehrere Zeilen parallel zu verarbeiten, das deutlich langsamste ist. Außerdem sind alle Modelle, die räumliche Daten verwerten langsamer als die Modelle B, C und D, die nur einzelne Spektren verarbeiten. Es ist kein Zusammenhang der Anzahl der Klassen mit der Dauer der Bewertung zu erkennen.
Bei dem Vergleich der Bewertungsdauer von vielen Zeilen auf einmal im Gegensatz zu wenigen Zeilen liegt der Faktor von Modell A wenig überraschend nahe Eins, da das Modell so oder so die Spektren einzeln verarbeiten muss. Bei den anderen Modellen liegt der Faktor zwischen 10 und 40 - meist aber bei etwa 30 -, wobei kein Zusammenhang mit variierten Parametern erkennbar ist. Es wäre möglich, dass diese Abweichungen an der variierenden Auslastung des PCs liegen könnten.

6.4 Erklärbarkeit der Entscheidungen und Einblick in die Modelle

6.4.1 Feature Importance

Bei den Random Forests kann die Feature Importance, wie in Kapitel 4.2 erläutert, direkt nach dem Training abgelesen werden. Bei neuronalen Netzen hingegen muss die Feature Importance nach dem Training errechnet werden. Sie kann zum Beispiel mit der Permutation Feature Importance (siehe Kapitel 4.5.2) bestimmt werden. Da diese Methode jedoch aufwendiger ist, wird sie nur für die neuronalen Netze, die nach den Korrektheitsanalysen die besten Ergebnisse geliefert haben, angewandt. Bei den

Random Forests hingegen wird die Feature Importance für alle Modelle ausgewertet. Zu beachten ist dabei jedoch, dass die Feature Importances der Random Forests nur bedingt mit denen der neuronalen Netze verglichen werden können, da sie über unterschiedliche Methoden berechnet wurden.

Random Forests

Bei den Random Forests zur Bewertung einzelner Spektren ähneln sich die Feature Importances der Modelle je Zielklasse wie beispielhaft in Abbildung 6.34 zu sehen. Dabei gilt, dass sich die Feature Importances mit - durch mehr Blätter oder Bäume - steigender Komplexität der Modelle immer weiter annähern. Die Modelle scheinen sich also mit steigender Komplexität ähnlicher zu werden, was für ein stabileres Training spricht. Das gilt für die Modelle aller drei Zielklassen.
Auch die Feature Importances der besten Random Forest Modelle je Zielklasse (siehe

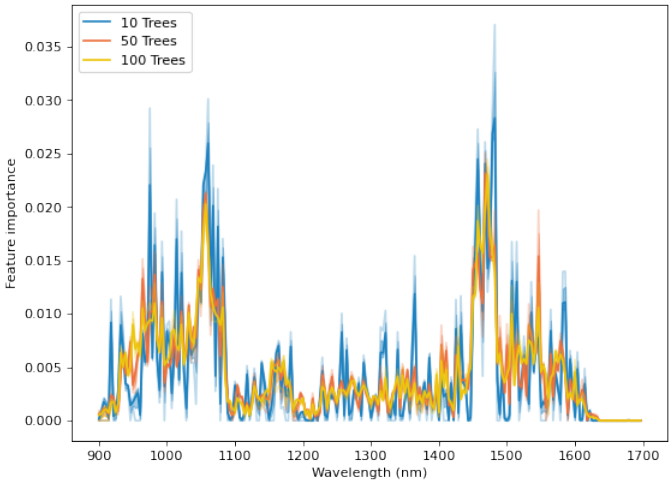


Abbildung 6.34: Mittelwert und Standardabweichung (heller hinterlegt) der Feature Importance der Random Forests aufgeteilt nach der Anzahl der Bäume

Abbildung 6.35) ähneln sich. Für alle Modelle gilt, dass es einige im Vergleich sehr wichtige Features gibt. Diese liegen bei etwa $1050nm$, $1150nm$ und $1470nm$. Dazwischen besitzen alle Modelle moderat wichtige bis vergleichsweise unwichtige Features. Besonders unwichtig bei allen Modellen ist der Wellenlängenbereich ab etwa $1600nm$, in dem das Spektrum durch schlechte Beleuchtung und fehlende Sensitivität der Kamera stark rauscht.
Die Basismodelle der Random Forests zur räumlichen und spektralen Bewertung entsprechen den rein spektralen Modellen. Daher entspricht auch die Feature Importance denen der jeweiligen spektralen Modelle. Bei dem Majority Classifier kann kein Unterschied der Wichtigkeit der Spektren in Abhängigkeit der Lage des zugehörigen Pixels festgestellt werden. Bei der Auswertung der Feature Importances des Majority Classifiers zur ausschließlichen Unterscheidung der Lacke sind die Basecoats ohne Klarlack jedoch etwas wichtiger für das Modell als die mit Klarlack lackierten Oberflächen. Gleiches gilt für den Majority Classifier, der zusätzlich in gelaserte und ungelaserte Oberflächen unterscheiden kann.
Bei der Feature Importance der Random Forests zur Unterscheidung in alle Klassen fallen regelmäßige Peaks in der Wichtigkeit auf. Bei der näheren Untersuchung

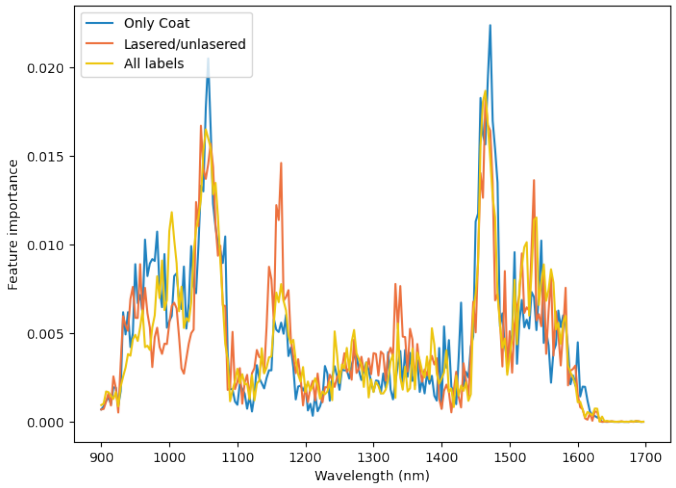


Abbildung 6.35: Feature Importance der in Kapitel 6.2 ausgewählten Random Forests je Zielklasse

stellt sich heraus, dass die Peaks zu den Klassen gehören, die für die ungelaserten Oberflächen stehen.

Neuronale Netze

Die Feature Importance der in Kapitel 6.2 ausgewählten neuronalen Netze ist in Abbildung 6.36 zu sehen. Zunächst fällt auf, dass sie viel kontinuierlicher ist als bei den Random Forests: Es gibt nur wenige sehr unwichtige Features und der Rest ist ähnlich wichtig. Unwichtig sind die Randbereiche der Wellenlängen, in denen das Signal stark rauscht.
Während die Feature Importances der Modelle zur ausschließlichen Unterscheidung der Lacke und zur Klassifizierung der Oberfläche in ungelasert und gelasert etwa gleich groß sind, ist die Feature Importance zur Differenzierung in alle Klassen deutlich niedriger. Das könnte dadurch zu erklären sein, dass der Fehler auf den originalen Daten deutlich höher ist als bei den anderen beiden Modellen, sodass der Fehler durch die Veränderung relativ zum initialen Fehler kleiner ist.

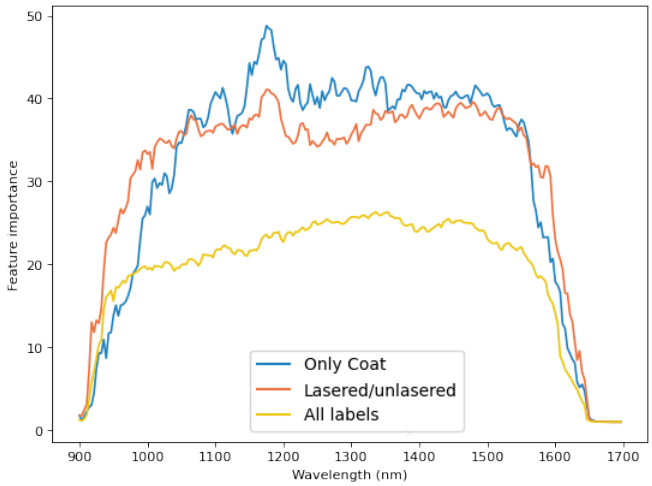
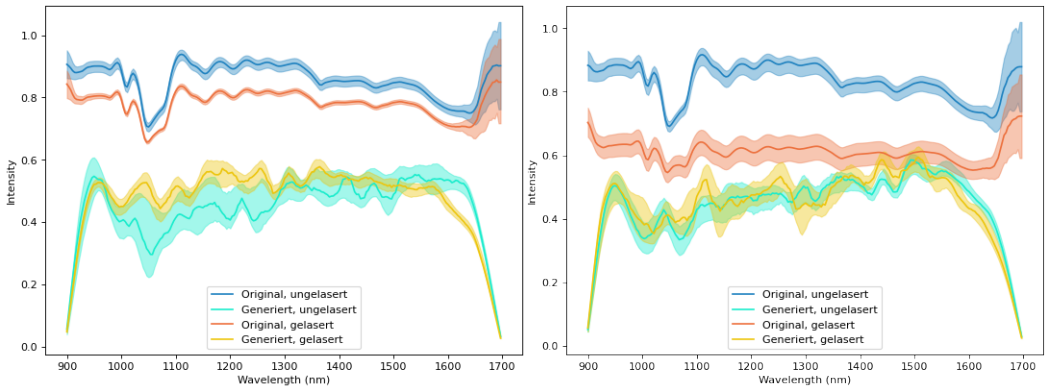


Abbildung 6.36: Feature Importance der in Kapitel 6.2 ausgewählten neuronalen Netze

6.4.2 Activation Maximization

Da die gefundenen Bibliotheken sich nicht für die Activation Maximization der Hypercubes eigneten, wurde der Algorithmus dafür selbst implementiert. Zunächst wird ein Spektrum mit zufälligen Werten initialisiert, aus dem ein Spektrum der ausgewählten Klasse erzeugt werden soll. Dann wird die Softmax-Schicht eines Modells entfernt, um das Problem der schwindenden Gradienten zu vermeiden. Als Zielvorhersage wird ein One-Hot-Vektor für die ausgewählte Klasse initialisiert, der statt einer Eins einen Wert von 100000 aufweist, um höhere Gradienten zu erzeugen. Das Modell klassifiziert das zufällig erzeugte Spektrum und als Loss wird die Summe der quadratischen Differenzen zwischen der Vorhersage und dem Zielvektor berechnet. Per Backpropagation wird dieser Fehler auf die Eingabe zurückgerechnet und das Eingabebild dementsprechend angepasst. Zur Regularisierung werden die Gradienten per L2 normiert.
Es wurde jedoch festgestellt, dass dieser Algorithmus zwar in der Lage ist, Spektren einer gewünschten Klasse zu erzeugen, diese aber keinerlei Ähnlichkeit mit den gemessenen Spektren haben. Stattdessen existieren hochfrequente, starke Änderungen zwischen benachbarten Wellenlängen, die das Signal wie starkes Rauschen wirken lassen. Um dieses zu kompensieren, wird das generierte Spektrum stets vor dem Bewerten durch das Modell per Gauss-Filter geglättet.

Zur Generierung der künstlichen Spektren per Activation Maximization wurde das Neuronale Netz zur Differenzierung in gelaserte und ungelaserte Oberflächen mit Modellarchitektur C ausgewählt. Es wurden jeweils zehn Spektren pro Klasse erzeugt. Für vier Klassen sind die Ergebnisse als Median mit hinterlegter Standardabweichung in Abbildung 6.37 dargestellt.
Alle künstlich erzeugten Spektren haben zwei Gemeinsamkeiten: Zum einen sinkt



(a) Alexit HS 411-22 KLM (b) Aviox CC UVR auf Alexit KLM

Abbildung 6.37: Median und Standardabweichung realer und künstlich generierter Spektren. Die Spektren wurden per Activation Maximization mit Modellarchitektur C des Netzes zur Differenzierung in gelaserte und ungelaserte Oberflächen erzeugt.

die Intensität des Spektrums an den Randbereichen des aufgenommenen Wellenlängenspektrums, in dem auch die Feature Importance des Modells niedrig ist, stark ab. Zum anderen haben alle eine deutlich ähnlichere Intensität als es die Originalspektren haben: Während bei den dunklen Lacken kaum Intensität gemessen werden konnte und andere Lacke deutlich mehr Licht reflektiert haben, generiert das ausgewählte

Netz Spektren, deren Intensität rund um 0.5 schwankt. Das Modell erkennt dementsprechend die Klassen nicht an den absoluten Intensitäten einzelner Wellenlängen. Die künstlich erzeugten Spektren haben je Klasse mal größere und mal niedrigere Abweichungen untereinander. Das Modell entwickelt jedoch stets ähnliche Spektren pro Klasse, während sich die Spektren der unterschiedlichen Klassen deutlich stärker voneinander unterscheiden.
Die Suche nach Ähnlichkeiten zwischen originalen und künstlich erzeugten Spektren fällt schwer: Während manchmal Tiefpunkte an ähnlichen Stellen liegen wie bei dem ungelaserten Spektrum bei etwa $1050nm$ in Abbildung 6.37a, hat das Modell an anderen Stellen ganz andere *Vorstellungen* eines idealen Verlaufs als an derselben Stelle für gelaserte Oberflächen.

6.4.3 Analyse der Random Forest Entscheidungen

Um auch bei den Random Forests einen Einblick in die Idealvorstellung eines Spektrums zu erhalten, werden die Grenzen für optimale Spektren errechnet und visualisiert. Dazu wird für eine ausgewählte Klasse das Blatt eines jeden Baums des Random Forests bestimmt, welches diese Klasse mit der höchsten Reinheit repräsentiert. Zu jedem dieser Blätter wird der Pfad bestimmt und die Bedingungen auf diesem Weg ausgewertet. Muss der Wert höher als ein Vergleichswert sein, so wird der Balken für diese Wellenlänge unter dem jeweiligen Wert grau gefärbt und bei Kleiner-Bedingungen umgekehrt. Dadurch entsteht ein Diagramm, indem ein mögliches Spektrum in den weißen Bereichen eingezeichnet werden kann. Je dunkler ein Bereich ist, desto mehr Knoten widersprechen einem Wert dieser Größe für die ausgewählte Klasse.

Abbildung 6.38 zeigt die Verteilung der Grenzen für den Alexit HS 411-22 KLM-Lack

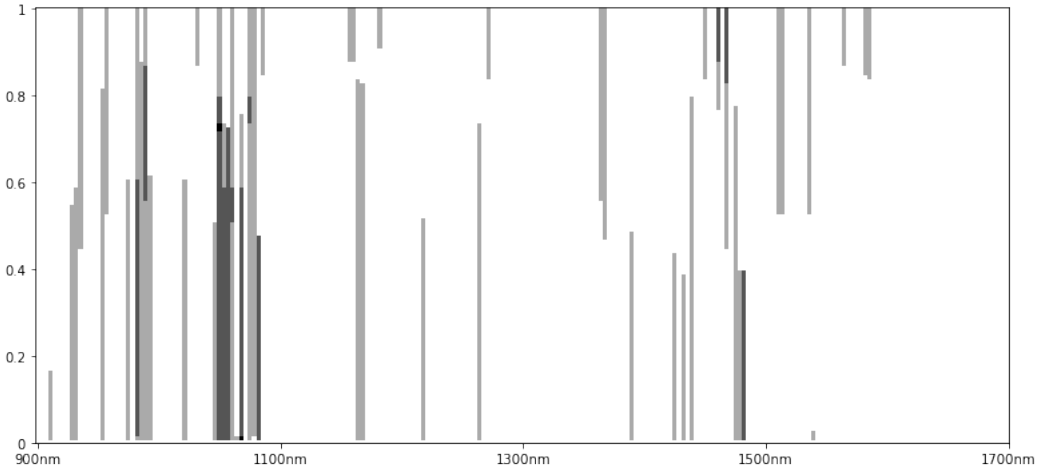


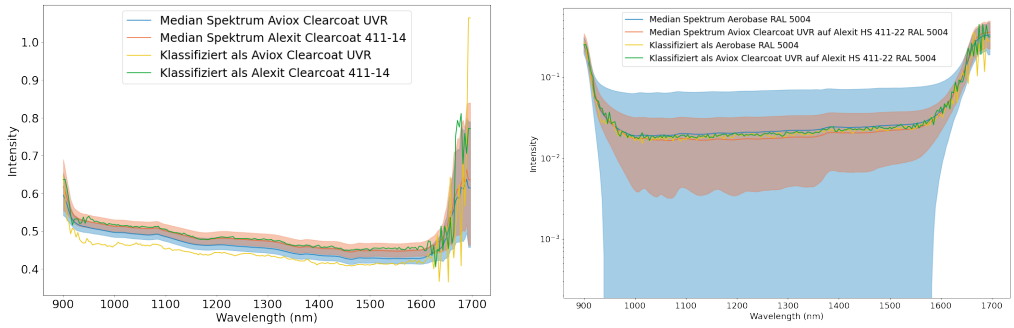
Abbildung 6.38: Visualisierung der Entscheidungen eines gesamten Random Forests zur Erkennung des Alexit HS 411-22 KLM

des Random Forests zur ausschließlichen Unterscheidung der Lacke mit 360 Blättern und 10 Bäumen an. Je dunkler ein Bereich dargestellt ist, desto mehr Knoten in den Bäumen widersprechen der Zugehörigkeit eines Spektrums, wenn die Wellenlänge die jeweilige Intensität hat. Dabei werden zwei Punkte deutlich: Wie schon die Feature-Importance gezeigt hat, werden stets wenige Wellenlängen zur Bewertung

der Oberflächen genutzt. Diese Grafik zeigt, dass dafür einige Wellenlängen mehrfach verwendet werden. Die mehrfach genutzten Wellenlängen stimmen mit denen überein, die nach der Feature Importance besonders wichtig für die Random Forests sind.
Außerdem zeigt die Grafik deutlich, dass sich die einzelnen Bäume teilweise widersprechen. Es gibt mehrere Wellenlängen, bei denen der graue Balken durchgehend ist, sodass in diesem Fall keine Intensität gewählt werden könnte, bei der alle Bäume des Random Forests maximal aktiviert werden.
Diese beiden Umstände treffen auf die Struktur der Visualisierungen anderer Klassen ebenfalls zu. Im Vergleich mit der Activation Maximization wird außerdem noch einmal deutlich, dass der Random Forest, so wie er trainiert wurde, nur die Möglichkeit hat, auf Basis der absoluten Intensitäten zu entscheiden.

6.4.4 Untersuchung der Spektren einiger fehlerhaft erkannter Pixel

Klassifikation durch Random Forest



(a) Spektren Kachel A

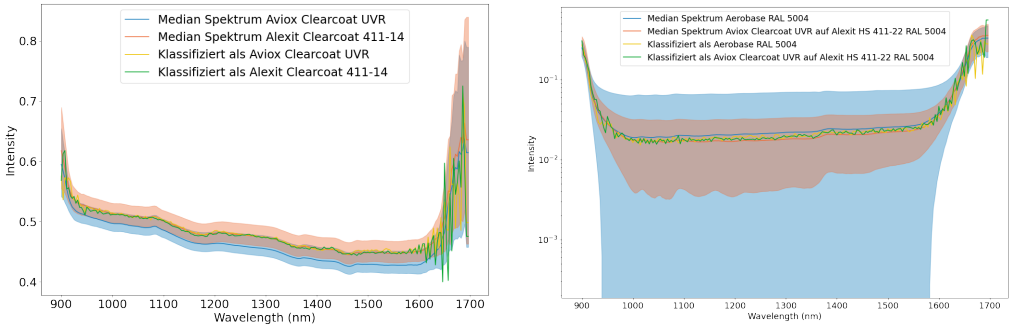
(b) Spektren Kachel B

Abbildung 6.39: Spektren jeweils zweier durch den Random Forest richtig und falsch klassifizierter Proben auf zwei unterschiedlichen Proben

In diesem Kapitel soll überprüft werden, ob in den Spektren offensichtliche Gründe für die Fehlklassifikation von Pixeln gefunden werden können. Dafür wird der nach den Korrektheitsanalysen als am besten geeignete Random Forest zur Differenzierung in gelaserte und ungelaserte Oberflächen verwendet.
Dazu wurden zwei Kacheln aus den Testdaten ausgewählt, von denen falsch klassifizierte Pixel mit korrekt klassifizierten Pixeln verglichen werden. Kachel A ist die in Abbildung 6.24 abgebildete Kachel, auf der grauer Basislack und Alexit Clearcoat 411-14 aufgetragen ist. Bei Kachel A wurde ein Pixel ausgewählt, auf dem das Modell zwar den richtigen Basislack, dafür aber den falschen Klarlack gewählt hat.
Kachel B stammt von einer Probe, auf die Aerobase RAL 5004 lackiert wurde. Das Modell hat das ausgewählte Pixel jedoch als Alexit HS 411-22 im Farbton RAL 5004 klassifiziert, der mit Aviox UVR überlackiert wurde.
In Abbildung 6.39 sind jeweils die mittleren Spektren des korrekten und des falsch klassifizierten Lacks sowie die Spektren der ausgewählten Pixel dargestellt.
Bei Kachel A hat das falsch klassifizierte, gelb eingezeichnete Spektrum im Vergleich zum richtig klassifizierten Spektrum und zum mittleren Spektrum eine niedrigere Intensität. Diese liegt näher an dem fälschlicherweise vorhergesagten Lack.
Für Kachel B wurden die Spektren mit logarithmischer Achse dargestellt, da sich zwischen den Spektren der beiden Lacke kaum Unterschiede zeigen. Trotzdem lässt sich keine Abweichung des Spektrums von den üblichen Verläufen erkennen, die eine falsche Vorhersage erklären würde.

Klassifikation durch Neuronales Netz

Zur Untersuchung der falschen Klassifikationen durch das neuronale Netz wurde wieder das Modell mit der Architektur C zur Differenzierung in gelasert und ungelasert verwendet. Die ausgewählten Kacheln entsprechen ebenfalls denen aus der vorangegangenen Auswertung. Lediglich die Pixel wurden entsprechend der Vorhersage neu ausgewählt.
In diesem Fall lassen sich jedoch keine Erklärungen für die falsche Vorhersage finden.



(a) Kachel A (b) Kachel B

Abbildung 6.40: Spektren jeweils zweier durch das neuronale Netz richtig und falsch klassifizierter Proben auf zwei unterschiedlichen Proben

Die ausgewählten Spektren von Kachel A sind nahezu gleich und liegen beide sehr nah an dem Median-Spektrum des korrekten Lacks. Bei der Kachel B ergibt sich ein ähnliches Bild wie bei der Auswertung der Vorhersage durch den Random Forest. Die Spektren der beiden Oberflächen sind sehr ähnlich und die Spektren der Pixel lassen sich nicht eindeutig zuordnen.

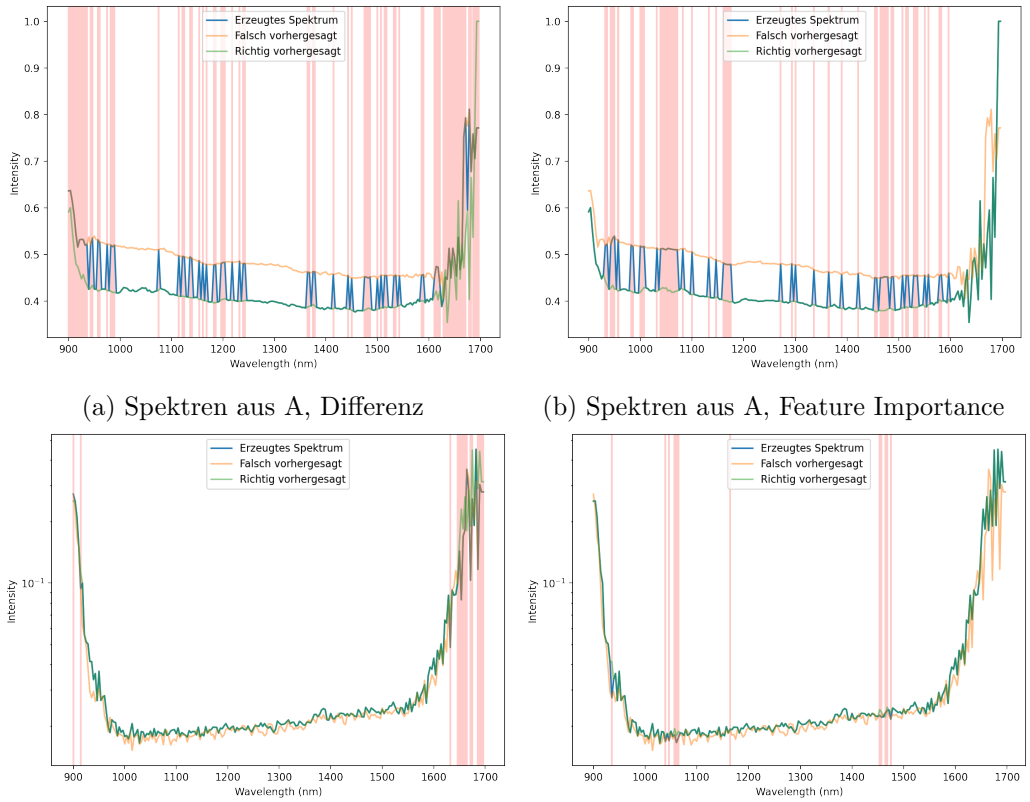
6.4.5 Counterfactual Explanations

Für die Berechnung der Counterfactual Explanations wurde zunächst die Bibliothek *Diverse Counterfactual Explanations (DiCE) for ML (dice-ml)* ausgewählt. Dabei bestand jedoch das Problem, dass die Berechnung sehr lange dauerte. Außerdem hatten die erzeugten Spektren an den veränderten Wellenlängen auffällige Peaks, die in den realen Daten so nicht vorkommen.
Daher werden die Counterfactual Explanations für diese Analysen stattdessen über die Annäherung eines falsch klassifizierten Spektrums an ein richtig eingeordnetes Spektrum über eine Heuristik berechnet. Dazu werden im ersten Schritt zwei Spektren ausgewählt, die zu der gleichen Oberfläche gehören, durch das jeweilige Modell aber unterschiedlich klassifiziert werden. Nach und nach werden die Werte einzelner Wellenlängen von dem richtig klassifizierten Spektrum auf das falsch erkannte Spektrums übertragen und das generierte Spektrum jeweils durch das Modell klassifiziert. Die Reihenfolge der Wellenlängen, die angepasst werden, wird über eine Heuristik ausgewählt. Dazu werden die größten Differenzen zwischen beiden Spektren und die Feature Importance des Modells verwendet, wobei als erstes die Wellenlängen mit den höchsten Differenzen bzw. der größten Feature Importance ausgewählt wird.

Random Forest

Für die Counterfactual Explanations der Random Forests wurde das Modell ausgewählt, welches bei der Korrektheitsanalyse am besten abgeschnitten hat. Es handelt

sich dabei um den Random Forest, der mehrere Spektren zur Klassifikation eines Spektrums nutzt und die Klassen in gelasert und ungelasert unterscheiden kann. Als Spektren, zu denen Counterfactual Explanations berechnet werden, wurden die aus Kapitel 6.4.4 verwendet.
In den Abbildungen 6.41a und 6.41b sind die nötigen Veränderungen bei dem ausge-



(a) Spektren aus A, Differenz

(b) Spektren aus A, Feature Importance

(c) Spektren aus B, Differenz

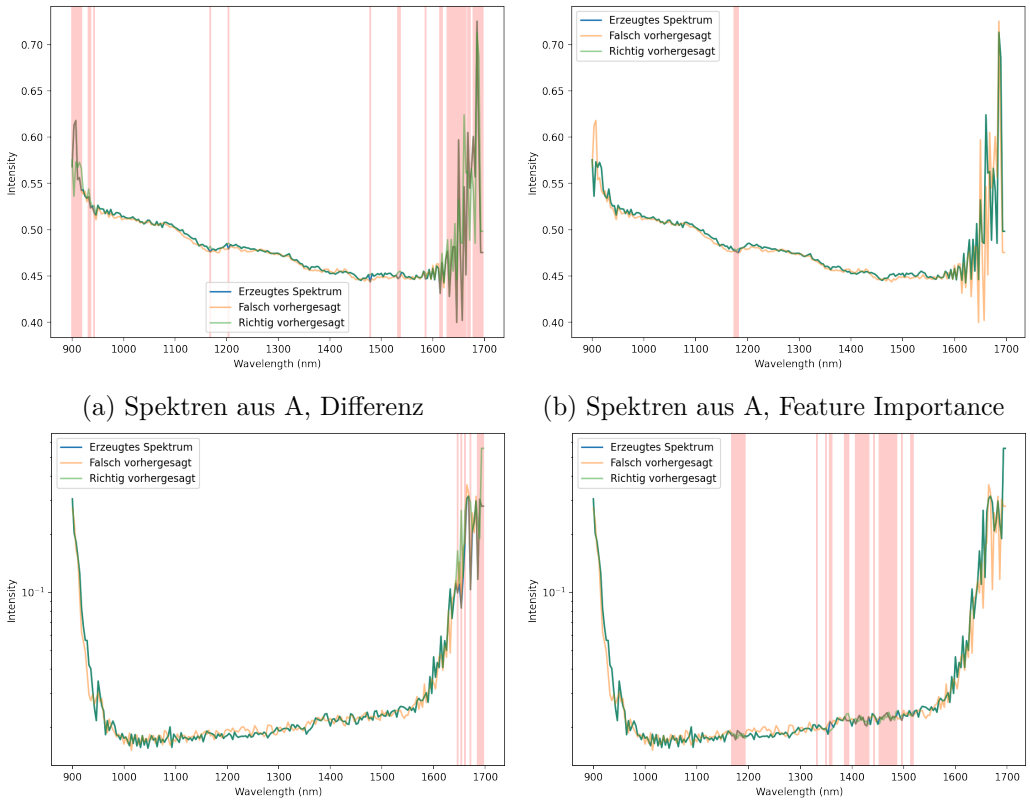
(d) Spektren aus B, Feature Importance

Abbildung 6.41: Counterfactual Explanations für die Klassifikation ausgewählter Spektren durch das beste räumliche Random Forest Modell zur Differenzierung in gelasert und ungelasert

wählten Spektrum, bei dem das Random Forest Modell den Klarlack verwechselt hat, dargestellt. Wie bereits im vorherigen Kapitel analysiert, hat das falsch klassifizierte Spektrum hier eine deutlich niedrigere Intensität als bei dieser Oberfläche üblich. Unabhängig von der Heuristik müssen hier die Werte vieler Wellenlängen verändert werden. Bei der Verwendung der größten Differenz sind es 80 und bei der Feature Importance 58.
Anders sieht es bei der Falschklassifikation auf Kachel B aus: Das korrekt klassifizierte Spektrum ist dem falsch klassifizierten Spektrum sehr ähnlich. Dementsprechend müssen weniger Werte verändert werden, um die Klassifikation zu korrigieren: Bei der Differenz sind 15 Veränderungen nötig und bei der Feature Importance 12.
Durch die sehr ähnlichen Verläufe der Spektren werden bei der Verwendung der größten Differenz zunächst die Randbereiche der Wellenlängen verändert, in denen das Signal rauscht. Bemerkenswert ist hier, dass das Modell seine Vorhersage korrigiert, obwohl diese Wellenlängen laut Feature Importance kaum genutzt werden.
Bei der Verwendung der Feature Importance sind die Änderungen sehr klein und verändern ebenfalls die Vorhersage.

Neuronales Netz

Auch beim neuronalen Netz wurde das Modell ausgewählt, welches bei den Analysen der Korrektheit am besten abgeschnitten hat. Es ist das Modell C, welches die Spektren einzeln verarbeitet und die Oberflächen in gelasert und ungelasert unterscheiden kann. Es wurden ebenfalls die in Kapitel 6.4.4 verwendeten Spektren zur weiteren Analyse ausgewählt.
Bei den Counterfactual Explanations zur Verwechslung des Klarlacks in den Abbil-



(a) Spektren aus A, Differenz

(b) Spektren aus A, Feature Importance

(c) Spektren aus B, Differenz

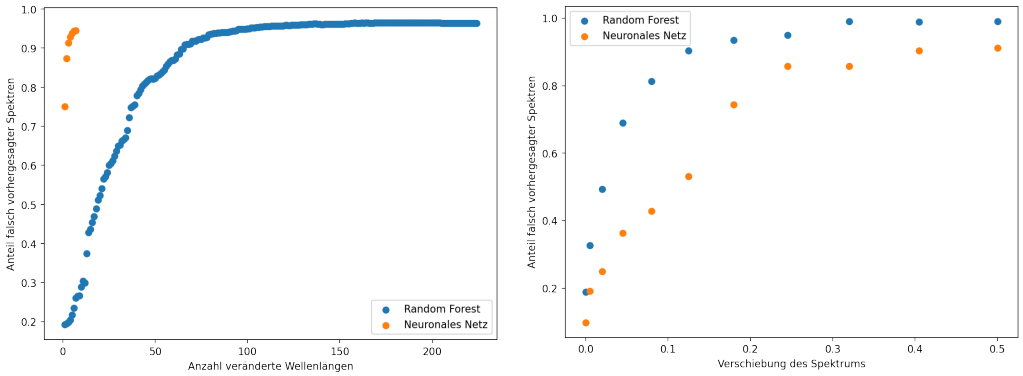
(d) Spektren aus B, Feature Importance

Abbildung 6.42: Counterfactual Explanations für die Klassifikation ausgewählter Spektren durch das neuronale Netz mit Architektur C zur Differenzierung in gelasert und ungelasert

dungen 6.42a und 6.42b fällt zunächst auf, dass bei der Verwendung der Feature Importance als Heuristik nur drei Wellenlängen angepasst werden müssen. Bei der Verwendung der Differenz werden mehr Wellenlängen angepasst. Darunter sind auch welche im Bereich der drei geänderten Wellenlängen. Es wäre also möglich, dass die großen Änderungen im verrauschten Bereich hier keinen oder kaum Einfluss auf die Bewertung durch das Modell haben.
Bei den Counterfactual Explanations zum Hinzuerfinden des Klarlacks in Kachel B (siehe Abbildung 6.42c und 6.42d) zeigt sich ein ähnliches Bild wie bei der Klassifizierung durch den Random Forest. Es reicht auch hier aus Wellenlängen, im verrauschten Bereich zu verändern, die laut Feature Importance kaum zur Bewertung hinzugezogen werden, aber anscheinend doch relevante Informationen enthalten können. Wird die Feature Importance verwendet, werden Wellenlängen angepasst, die in beiden Spektren sehr ähnlich sind. Dadurch müssen deutlich mehr Werte verändert werden.

6.4.6 Robustheit der Modelle gegenüber Veränderungen der Spektren

Abschließend soll untersucht werden, wie empfindlich die Modelle auf Änderungen reagieren. Dazu werden Versuche mit zwei Arten von Veränderungen durchgeführt: Zunächst werden die einzelnen Wellenlängen der Wichtigkeit nach verändert und dabei überwacht, wie viele Spektren noch korrekt klassifiziert werden. Im anderen Fall werden die Spektren um einen konstanten Betrag verschoben. Für beide Versuche werden die Spektren des Testdatensatzes verwendet.
Als Modell wird hier wieder das Neuronale Netz der Modellarchitektur C zur Differenzierung der Oberflächen in gelasert und ungelasert verwendet. Um ein vergleichbares Ergebnis zu erhalten, wird dieses Mal allerdings der rein spektrale Random Forest verwendet, der ebenfalls gelaserte von ungelaserten Oberflächen unterscheiden kann. In Abbildung 6.43 sind die Ergebnisse abgebildet. Bei der zufälligen Veränderung



(a) Zufällige Veränderung einzelner Wellenlängen

(b) Verschiebung des gesamten Spektrums

Abbildung 6.43: Anteil der nach Veränderung der Spektren noch richtig vorhergesagten Daten

einzelner Wellenlängen ist der Random Forest deutlich robuster als das neuronale Netz. Das neuronale Netz reagiert bereits sensibel auf eine veränderte Wellenlänge, weswegen der Versuch nach wenigen Wellenlängen abgebrochen wurde. Auch der Random Forest wird durch die Veränderung deutlich unsicherer. Beim Verändern weniger Wellenlängen kann er die Klasse jedoch trotzdem noch gut bestimmen. Bei den neuronalen Netzen fällt auf, dass sie bei der Veränderung einzelner Wellenlängen dazu tendieren, die Undefined-Klasse vorherzusagen, die bei realen Spektren nicht vorhergesagt wird.
Bei der Verschiebung ganzer Spektren ergibt sich ein anderes Bild: Hier zeigt sich, dass das neuronale Netz dadurch weniger beeinflusst wird. Das neuronale Netz ist stets mindestens um den initialen Unterschied besser als der Random Forest. Insbesondere im Bereich einer Verschiebung um 0.05 bis 0.2 ist das neuronale Netz deutlich weniger empfindlich.

7 Diskussion der Untersuchungsergebnisse

Um das am besten geeignete Modell auszuwählen, werden die Analyseergebnisse aggregiert, bewertet und gegeneinander abgewägt.
Einige Modelle lassen sich durch die Ergebnisse der Geschwindigkeitsanalyse schnell ausschließen. Hier ist die Vorgabe, dass die Zeilen mit mindestens $50Hz$ aufgenommen werden sollen. Optimal wäre es, wenn das Modell eine Framerate von $200Hz$ verarbeiten könnte, da dies die maximale Aufnahmefrequenz der Kamera unter den übrigen gewählten Parametern ist. Auch zum Abschluss dieser Arbeit steht noch nicht fest, wie viele Zeilen parallel verarbeitet werden können. Die Tendenz geht aber dahin, dass der Laser nicht der Kamera direkt folgt, um die Kamera zu schützen. Stattdessen wird ein Werkzeugwechsel nötig sein. Das heißt, dass das Modell trotzdem Zeilen mit $50Hz$ bis $200Hz$ verarbeiten können muss, es aber dafür mehrere Zeilen zu einem Bild zusammenfassen kann. Für die Klassifizierung von 1000 Zeilen, stehen also $5s$ bis $20s$ zur Verfügung.
Damit scheidet die Architektur des neuronalen Netzes A, die aus der Literatur stammt, aus. Bei den Random Forests würden sich die Modelle zwar alle für eine Framerate von $50Hz$ eignen, wenn 1000 Zeilen zusammengefasst werden, höhere Frequenzen funktionieren aber nur mit wenigen Schätzern pro Random Forest. Passenderweise hat sich dort ebenfalls gezeigt, dass das Verwenden von mehr als zehn Bäumen keinen oder kaum positiven Einfluss hat. Auf der Basis dieser Analysen sollten Random Forests mit mehr als zehn Bäumen für diese Aufgabe nicht verwendet werden.
Die Random Forests mit maximal zehn Prädiktoren, die einzelne Spektren betrachten, eignen sich sogar für die Klassifikation von je drei Zeilen parallel, dann allerdings je nach Zielklasse nur noch bis maximal 100 Hz. Die Random Forests, die zu den spektralen Informationen auch räumliche verarbeiten, sind deutlich langsamer und eignen sich nicht für die iterative Klassifikation weniger Zeilen.
Derselbe Umstand trifft auf die neuronalen Netze zu: Bei ihnen ist der Geschwindigkeitsunterschied zwischen der Klassifikation einzelner Zeilen und ganzer Bilder noch größer, sodass die Zielgeschwindigkeiten bei der Verarbeitung weniger Zeilen parallel nicht erreicht werden können. Auch hier sind die Modelle zur rein spektralen

Betrachtung schneller und eignen sich für Aufnahmefrequenzen bis $200Hz$. Die Netze, die auch räumliche Informationen verarbeiten, kommen hingegen maximal auf etwa $100Hz$. Dafür ist bei den neuronalen Netzen die Geschwindigkeit im Gegensatz zu den Random Forests unabhängig von den Zielklassen.
Sollten die Anforderungen also dahingehend verschärft werden, dass einzelne Zeilen verarbeitet werden, eignen sich nur Random Forests dafür. Kann die Anzahl der parallel verarbeiteten Zeilen hingegen weitgehend frei gewählt werden, so kann ein beliebiger Random Forest mit 10 Bäumen oder eines der neuronalen Netze außer Modell A gewählt werden und mit den meisten dieser Modelle wird sich eine Verarbeitungsfrequenz von $200\frac{\text{Zeilen}}{s}$ erreichen lassen.

Das nächste Kriterium für die Auswahl eines Modells ist die Güte dessen Klassifikation. Hier konnten mit der Korrektheitsanalyse detaillierte Einblicke gewonnen werden. Bei den neuronalen Netzen hat sich gezeigt, dass eine Verarbeitung von räumlichen Informationen keinen Mehrwert bringt. Da diese zudem langsamer sind, können sie aus der Auswahl ausgeschlossen werden.
Bei den Random Forests ergab sich, dass eine höhere Anzahl der Blätter einen positiven Einfluss auf die Bewertung hat. Die Metriken lassen vermuten, dass eine weitere Erhöhung dieser Anzahl den positiven Trend fortsetzen würde. Da die Dauer für die Klassifizierung nur logarithmisch in die Geschwindigkeit eingeht, könnte hier eine Erhöhung zu noch besseren Modellen führen.
Gibt es nur geringe Anforderungen an die Geschwindigkeit und können mehrere Zeilen auf einmal verarbeitet werden, so empfiehlt sich die Nutzung der Random Forests, die mehrere Spektren auf einmal räumlich verarbeiten, da sie weniger Fehler machen.
Eine wichtige Erkenntnis der Korrektheitsanalysen war, dass die Metriken für diese Anwendung keinen direkten Aufschluss über die Eignung eines Modells liefern. Ein Modell mit niedrigeren Metriken kann besser sein als eines, das sehr gute Ergebnisse liefert. Dieses liegt an der unterschiedlichen Gestaltung der Zielklassen: Ein Modell, das die Anzahl der Laserzyklen bestimmen soll, muss eine deutlich schwierigere Aufgabe erledigen als eines, das nur den Lack bestimmen soll. Daher sind die Metriken des ersten deutlich schlechter. Werden die Arten der Fehler analysiert, so zeigt sich aber, dass der Großteil der Fehler in der falschen Anzahl der Laserzyklen liegt und die Modelle tendentiell weniger oft den Lack verwechseln. Daher sollte unabhängig davon, ob die Daten zum Laserzustand der Oberflächen benötigt werden oder nicht, ein Modell gewählt werden, das die Laserzustände berücksichtigt.
Die aktuellen Ergebnisse der Untersuchungen des Laserns deuten darauf hin, dass der Worst Case eine Fehlklassifizierung des Basislacks ist. Wenn ein nicht vorhandener Klarlack erkannt wird oder ein aufgetragener Klarlack übersehen wird, kann dies ebenfalls Probleme verursachen. Welcher Klarlack aufgetragen wurde, ist aber zweitrangig. Das neuronale Netz zur rein spektralen Auswertung und Unterscheidung in gelaserte und ungelaserte Zustände liefert hier die besten Ergebnisse und liegt nur bei etwa 1% der Beispiele mit seiner Vorhersage für den Basislack falsch. Der beste Random Forest führt hier zu etwa 2% falscher Klassifikation.
Analysen von detaillierten Aufnahmen konnten zeigen, dass es einige Pixel gibt, in denen das Modell falsch liegt, dass es aber auch Bereiche gibt, in denen die Probe nicht genau genug annotiert wurde. Daher wird die tatsächliche Rate dieser Art von Fehler unter 1% bzw. 2% liegen. Dazu kommt, dass die Probenbleche Krümmungen aufweisen, die an realen Bauteilen nicht zu erwarten sind. An dieser Stelle wird die

Reflektion des infraroten Lichts verändert, sodass sich die Spektren verändern und die Modelle es schwerer haben. Trotzdem waren die neuronalen Netze in der Lage, diesen Effekt etwas zu kompensieren.
Weitere Fehlklassifikationen des Basislacks entstehen häufig bei den dunklen Lacken. Trotz der niedrigen Intensität und dadurch sehr ähnlichen Spektren können die trainierten Modelle und insbesondere die neuronalen Netze diese Klassen relativ zuverlässig auseinander halten.
Wird ein Spektrum trotzdem falsch klassifiziert, so lässt sich trotz der Verwendung von Methoden der erklärbaren KI kaum sagen, woran es liegt. Dies liegt allerdings auch daran, dass die Zusammensetzung des Spektrums nicht einfach nachzuvollziehen ist. Vor der Aufnahme des Spektrums einer neuen Oberfläche lässt sich maximal die gesamte Intensität anhand der Farbe schätzen, hingegen nicht welche Wellenlängen die Oberfläche mehr oder weniger reflektiert. Daher können menschliche Beobachter maximal die Ähnlichkeit zu einem oder mehreren Referenzspektren beurteilen. Dieses Verfahren kommt jedoch an seine Grenzen, wenn es wie bei den dunklen Farbtönen und darauf aufgetragenen Klarlacken mehrere, sehr ähnliche Spektren gibt. In einigen Fällen scheinen auch die Modelle diese Probleme zu haben: Die Counterfactual Explanations haben gezeigt, dass manchmal die Veränderung sehr weniger Werte dazu führt, dass sich die Klassifikation durch das Modell ändert. Andere Counterfactual Explanations zeigten jedoch, dass viele Features angepasst werden müssen, damit die Klasse geändert wird. Allerdings wurden die Counterfactual Explanations nicht per Optimierung sondern per Heuristik erstellt. Counterfactual Explanations, die z.B. ähnlich wie bei der Activation Maximization per Backpropagation und Gradientenabstieg erstellt werden, könnten andere Ergebnisse liefern.
Dieser Umstand - also die vielen, schwer begreifbaren Features - erschwert auch die Erklärbarkeit der Modelle. Trotzdem konnten einige Einblicke gewonnen werden: Die erste Beobachtung wurde bereits bei der Geschwindigkeitsanalyse gemacht: Bäume von Random Forests, die mehr Klassen zu unterscheiden haben, sind tiefer als Modelle mit gleich vielen Blättern, die aber weniger Klassen differenzieren müssen. Warum die Modelle so erstellt werden, steht jedoch noch nicht fest.
Bei der Analyse der Spektren hat sich gezeigt, was die Daten der NIR-Beleuchtung schon vermuten ließen: Die Beleuchtungsintensität ist bei den niedrigen und hohen Wellenlängen eher gering. Dadurch sinkt die Intensität und das Rauschen nimmt stark zu. Die Modelle sollten also lernen, dass der Informationsgehalt dieser Bereiche niedriger ist. Die Analyse der Feature Importances ergibt, dass dies für alle Modelle zutrifft. Einige der Counterfactual Explanations zeigen jedoch, dass wenn die Spektren sehr ähnlich zueinander sind, das Rauschen trotzdem zur Differenzierung verwendet werden kann.
Die Feature Importance legt außerdem den Schluss nahe, dass der Random Forest nur wenige Wellenlängen zur Entscheidung berücksichtigt. Dieses wird durch die Analyse der Entscheidungspfade bestätigt. Im Gegensatz dazu scheinen die neuronalen Netze zur Klassifikation die meisten Wellenlängen etwa gleich ähnlich zu berücksichtigen. Dieses könnte vermuten lassen, dass die neuronalen Netze robuster gegenüber Störungen sind, was jedoch nicht bestätigt werden konnte. Bei den neuronalen Netzen können bereits wenige Wellenlängen, deren Werte durch zufällige Werte ersetzt werden, zur Falschklassifikation führen. Allerdings fällt dabei auf, dass das neuronale Netz diese Daten als nicht *normal* klassifiziert und zum großen Teil in die Klasse *Undefined* einsortiert, obwohl es nicht darauf trainiert wurde, *Undefined* zu erkennen. Eine Möglichkeit für das Nichtzutreffen der Vermutung, dass die neuronalen Netze

robuster sind, ist, dass die Feature Importances der Neuronalen Netze nicht mit denen der Random Forests vergleichbar sind, da sie nach unterschiedlichen Methoden errechnet wurden.
Mit Hilfe der Activation Maximization konnten Musterspektren für das neuronale Netz erstellt werden. Vor allem ohne eine Glättung aber auch nach einer Filterung haben diese jedoch kaum Ähnlichkeit mit den aufgenommenen Spektren. Das neuronale Netz scheint also weder auf den offensichtlichen, groben Verlauf, noch auf die Intensität, sondern auf Feinheiten oder andere versteckte Features zu achten. Stattdessen zeigt sich, dass die Intensität der generierten Spektren aller Klassen etwa gleich hoch ist. Dieses führt zu der Vermutung, dass die neuronalen Netze nicht auf die Intensitäten achten. Für eher niedrigere Verschiebungen konnte sich diese Vermutung im Gegensatz zu den Random Forests bestätigen. Bei höheren Offsets wird das Spektrum jedoch auch aus dem üblichen Wertebereich herausgeschoben, was wiederum für negative Effekte sorgen kann. Dass die neuronalen Netze etwas unempfindlicher gegenüber Verschiebungen sind, ist dahingehend positiv, dass eine Veränderung der Beleuchtung oder Belichtungsdauer zu einer Verschiebung führen würde, mit der ein neuronales Netz dann tendenziell weniger Probleme haben sollte.
Aus allen diesen Analysen ergibt sich das Fazit, dass ein Modell gewählt werden sollte, das die Oberfläche in gelasert und ungelasert differenzieren kann. Können mehrere Zeilen auf einmal verarbeitet werden, so sollte ein neuronales Netz verwendet werden, da dieses die besten Ergebnisse liefert und etwas robuster gegen Veränderungen der Intensität ist. Nur wenn es unbedingt nötig ist, sollten die aufgenommenen Zeilen einzeln verarbeitet werden, da sich in diesem Fall nur die rein spektralen Random Forests eignen, die jedoch weniger genaue Ergebnisse liefern als die anderen Modelle. Für sie sollten in diesem Fall 10 Bäume und möglichst viele Blätter verwendet werden.

8

Zusammenfassung und Ausblick

Es konnten mehrere Modelle identifiziert werden, die in der Lage sind„ die gestellten Anforderungen zu erfüllen und Lackschichten mit etwa 90% Genauigkeit zu bestimmen. Basislacke werden sogar mit bis zu 99% Trefferquote erkannt, obwohl einige Lacke die gleiche Farbe haben, mit dem Auge nicht zu unterscheiden sind und teilweise durch ihre dunkle Farbe nur sehr wenig Licht reflektieren.
Für die Entwicklung eines geeigneten Systems wurde zunächst ein Datensatz erstellt und eine Auswahl an Vorverarbeitungsschritten und Modellen getroffen. Diese Modelle wurden in vielen verschiedenen Konfigurationen trainiert und ihre Ergebnisse ausgewertet. Je nach Anforderungen werden dabei Neuronale Netze oder Random Forests, die ausschließlich spektrale Informationen verarbeiten, empfohlen. Es wurde festgestellt, dass es den Modellen helfen kann, wenn die Klassen ausdifferenziert werden, auch wenn sie es dadurch schwerer haben, die korrekte Klasse zu wählen, was sich in schlechteren Metriken zeigt.
Die Korrektheitsanalysen der Random Forests ergeben, dass sich eine weitere Erhöhung der Anzahl der Blätter positiv auswirken kann. Bei den neuronalen Netzen konnte die Geschwindigkeit der Vorhersage von der Architektur aus der Literatur um den Faktor 30 erhöht werden.
In der Literatur wurden vor allem für die neuronalen Netze weitere Techniken zur Optimierung vorgestellt. Dazu gehören zum Beispiel Autoencoder und Regularisierung per Dropout oder Batch Normalisierung. Da die Analysen in dieser Arbeit bereits in mehreren Dimensionen recht umfangreich waren, wurde darauf verzichtet. Es wäre jedoch sehr interessant, ob die Modelle damit noch weiter verbessert werden könnten und z.B. Dropout die neuronalen Netze robuster werden lassen würde.
Mit Methoden der erklärbaren KI und genauen Analysen der Klassifikationen konnten einige Einblicke in die Modelle und Anhaltspunkte für die Entscheidungen gewonnen werden. Trotzdem ist die Entscheidung nach wie vor nicht interpretierbar. Alle Modelle sind gleich gut oder schlecht erklärbar, was allerdings zum großen Teil auch an der schweren Begreifbarkeit der Daten liegt. Mit weiteren Algorithmen der Erklärbarkeit könnten weitere Einblicke gewonnen werden. Viele dieser Methoden sind jedoch auf weniger Features ausgelegt, deren semantische Bedeutung dafür klarer ist.

Während der Analysen sind Fragen aufgetaucht, die bisher nicht beantwortet werden konnten. Dazu gehören: *Warum sind Entscheidungsbäume, die mehr Klassen differenzieren müssen, bei gleicher Blattanzahl tiefer? Lassen sich Muster in den vorhergesagten Klassen bei der Veränderung von Spektren erkennen?* Auch hier sind weitere Analysen nötig, um diese Fragen beantworten zu können.
Aber auch ohne die vorgeschlagenen Verbesserungen und Analysen eignen sich die Modelle bereits, um Informationen für den Laser bereitzustellen. Dafür ist jedoch noch einiges an Arbeit nötig: Der erste Schritt wäre, die Software in einem Live-System zu vereinen, das in der Lage ist, Bilder aufzunehmen und direkt zu klassifizieren. In einem weiteren Schritt müssten die Ergebnisse auf die Oberfläche gemappt werden, damit der Laser entsprechend programmiert werden könnte. Für einen Einsatz wäre außerdem die einfache Erweiterbarkeit um mehr Lacke praktisch. Da auch bisher von den meisten Lacken nur eine Probe für das Training verwendet wurde, sollte dies zunächst kein Problem darstellen. Sollen mehr als 256 Klassen erkannt werden, müsste allerdings die Speicherung der Annotationen in 8-Bit-Bildern angepasst werden.
Vor einem Realeinsatz sollten die Ergebnisse jedoch noch mit gealterten Oberflächen verifiziert werden, da die Aufnahmen für das Training der Modelle nur von frisch lackierten Proben gemacht wurden.

Anhang

Hersteller	Name	Typ	Farbe
AkzoNobel	Aerodur HS 2121 CF Primer	Primer	weiß
Mankiewicz	Alexit H/S Basecoat 411-22	Basecoat	grau
Mankiewicz	Alexit H/S Basecoat 411-22	Basecoat	gelb
Mankiewicz	Alexit H/S Basecoat 411-22	Basecoat	schwarz
Mankiewicz	Alexit H/S Basecoat 411-22	Basecoat	KLM
Mankiewicz	Alexit H/S Basecoat 411-22	Basecoat	RAL 5004
AkzoNobel	Aerobase	Basecoat	RAL 5004
AkzoNobel	Aerobase	Basecoat	blau
Maniewicz	Alexit Clearcoat 411-14	Clearcoat	
AkzoNobel	Aviox Clearcoat UVR	Clearcoat	
Maniewicz	Alexit H/S Topcoat 411-77	Topcoat	schwarz
AkzoNobel	Aviox 77702 Finish	Topcoat	grau

Tabelle 1: Liste der verwendeten Lacke

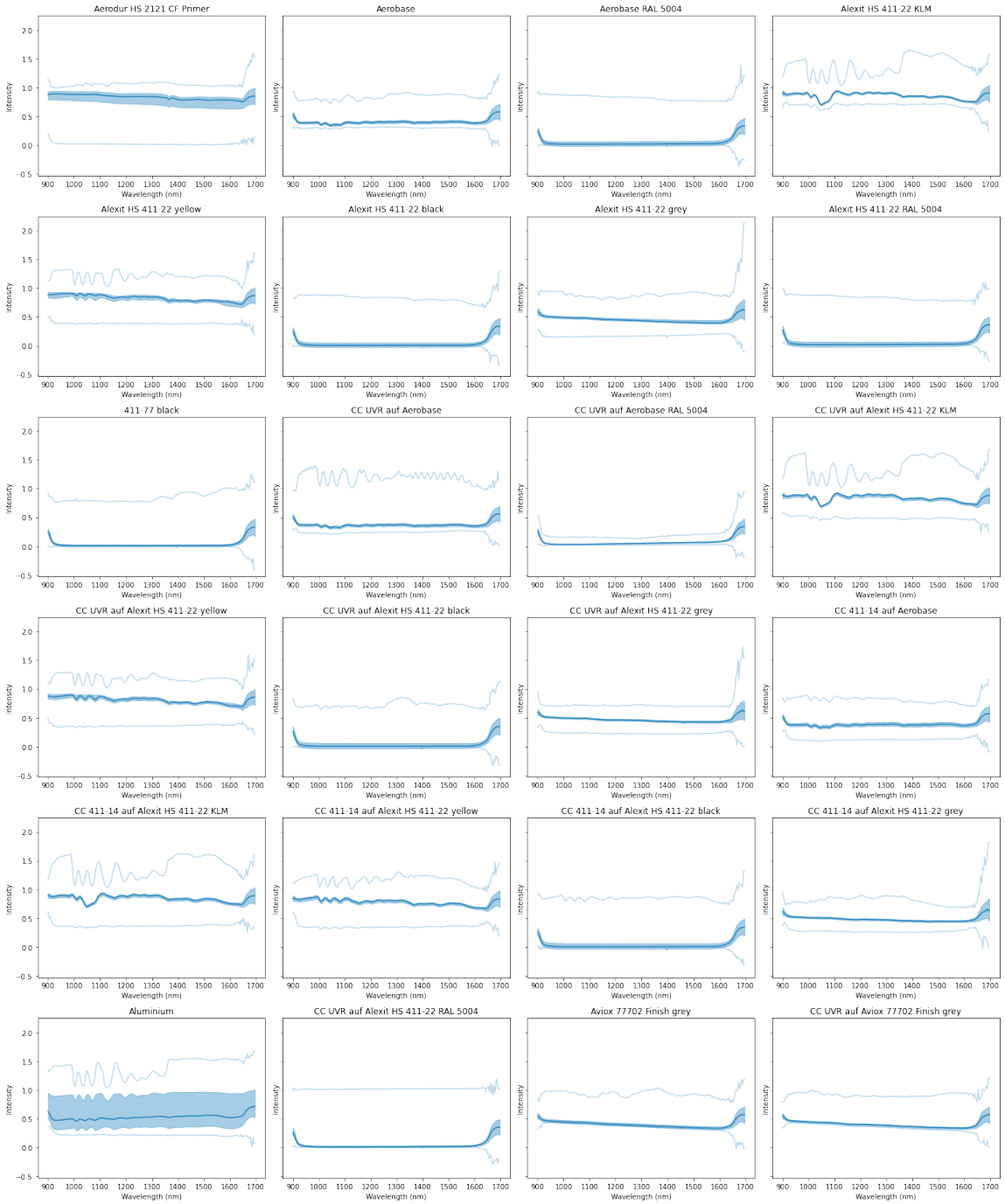


Abbildung 1: Median der Spektren der einzelnen ungelaserten Klassen mit Standardabweichung (mittelblau) und Extremwerten (hellblau)

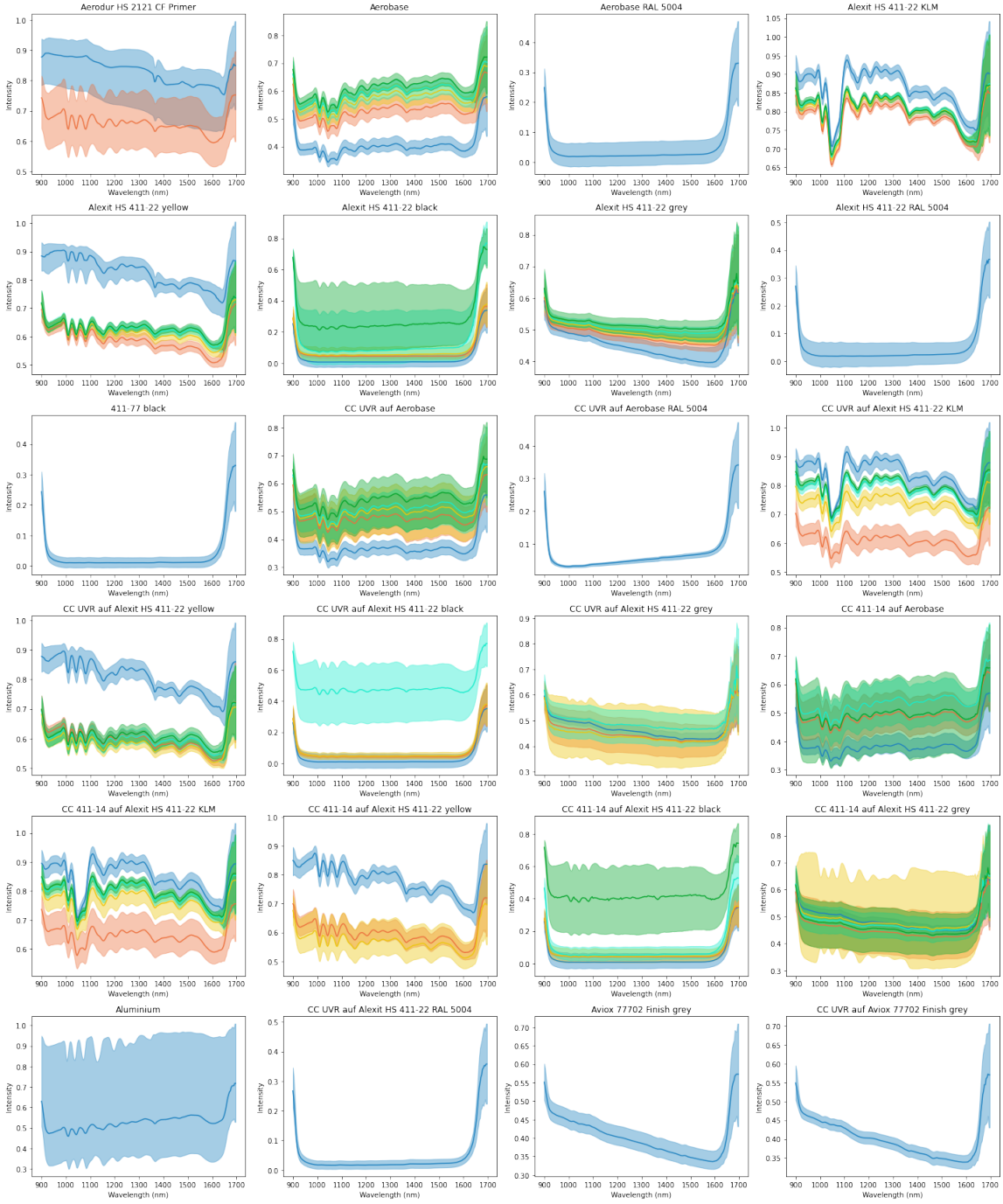


Abbildung 2: Median der Spektren der Klassen mit Standardabweichung aufgeteilt nach Laserzyklus (0 Zyklen: blau; 1 Zyklus: orange; 2 Zyklen: gelb; 3 Zyklen: türkis; 4 Zyklen: grün)

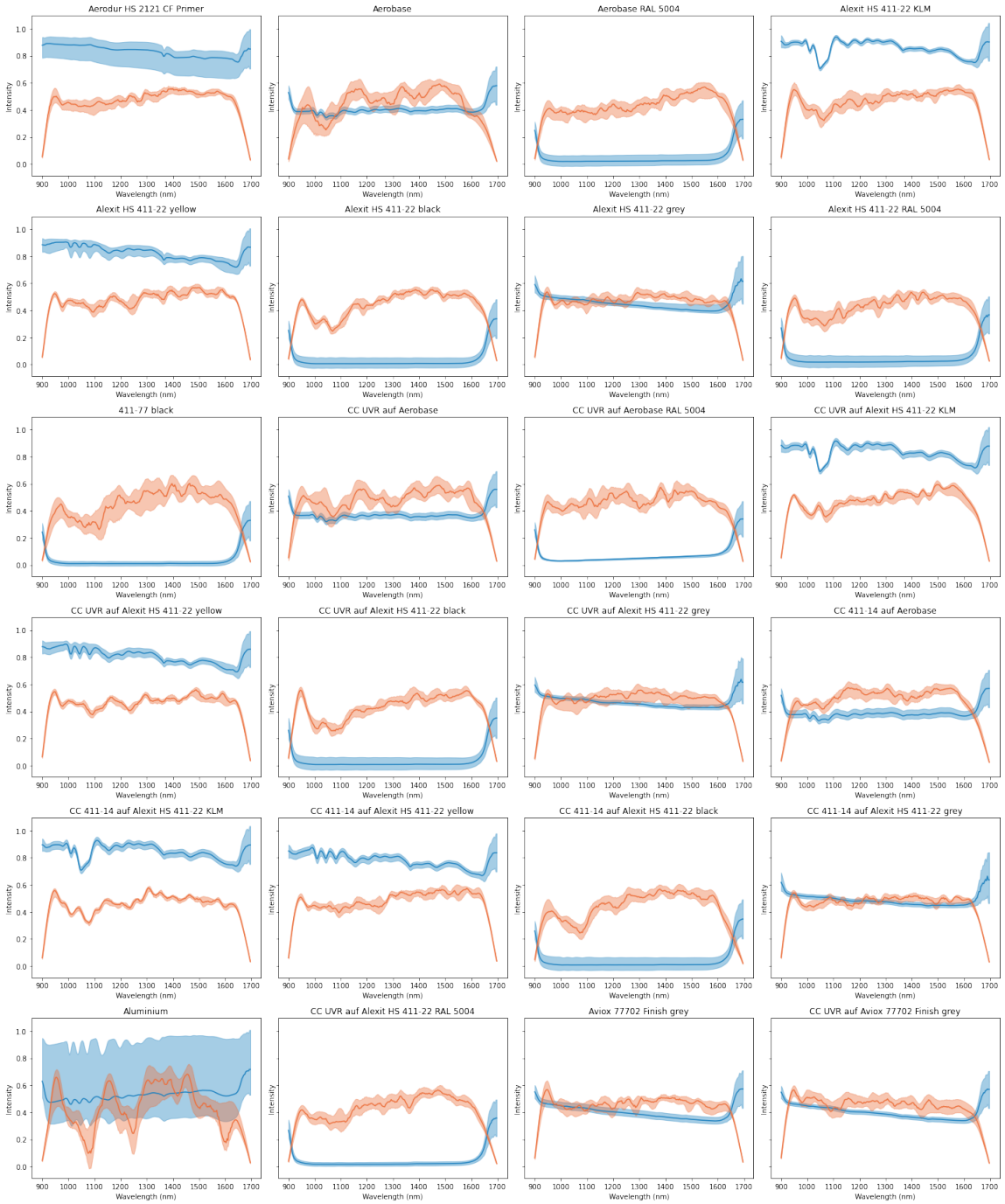


Abbildung 3: Median und Standardabweichung der originalen (blau) und durch Activation Maximization generierten (orange) Spektren

Abkürzungsverzeichnis

CART Classification and Regression Trees
CNN Convolutional Neural Network
DBN Deep Belief Network
FN False Negative
FP False Positive
FPS Frames per Second
GUI Graphical User Interface
ID Identifikationsnummer
ID3 Iterative Dichotomiser 3
JSON JavaScript Object Notation
IFAM Institut für Fertigungstechnik und Angewandte Materialforschung
KI Künstliche Intelligenz
LED Leuchtdiode
NIR Nahinfrarot
px Pixel
PNG Portable Network Graphics
RAL siehe RAL-Farbsystem
ReLU Rectified Linear Unit
RFE Recursive Feature Elimination
RGB Rot, Grün, Blau
ROI Region of interest
SAE Stacked Autoencoder
SVM Support Vector Machine
SWIR Infrarotes Licht; $1000 - 2500\ nm$
TN True Negative
TP True Positive
UUID Universally Unique Identifier

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] José Manuel Amigo. Chapter 1.1 - Hyperspectral and multispectral imaging: setting the scene. In José Manuel Amigo, editor, *Hyperspectral Imaging*, volume 32 of *Data Handling in Science and Technology*, pages 3–16. Elsevier, 2020.

[2] Muhammad Saad Shaikh, Keyvan Jaferzadeh, Benny Thörnberg, and Johan Casselgren. Calibration of a hyper-spectral imaging system using a low-cost reference. *Sensors*, 21(11), 2021.

[3] aprentas, editor. *IR-Spektroskopie*, pages 157–180. Springer International Publishing, Cham, 2017.

[4] Ian J. Maybury, David Howell, Melissa Terras, and Heather Viles. Comparing the effectiveness of hyperspectral imaging and Raman spectroscopy: a case study on Armenian manuscripts. *Heritage Science*, 6(1), 2018.

[5] Paul Geladi, Jim Burger, and Torbjörn Lestander. Hyperspectral imaging: calibration problems and solutions. *Chemometrics and Intelligent Laboratory Systems*, 72(2):209–217, 2004. Advances in Chromatography and Electrophoresis - Conferentia Chemometrica 2003, Budapest.

[6] José Manuel Amigo and Carolina Santos. Chapter 2.1 - preprocessing of hyperspectral and multispectral images. In José Manuel Amigo, editor, *Hyperspectral Imaging*, volume 32 of *Data Handling in Science and Technology*, pages 37–53. Elsevier, 2020.

[7] Spectral Imaging Ltd. Specim. Smile and keystone. https://www.specim.fi/smile-and-keystone/. Abgerufen am 10.01.2022.

[8] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers and Electrical Engineering*, 40(1):16–28, 2014. 40th-year commemorative issue.

[9] Wenjing Lv and Xiaofei Wang. Overview of hyperspectral image classification. *Journal of Sensors*, 2020:1–13, 07 2020.

[10] Rick Archibald and George Fann. Feature selection and classification of hyperspectral images with support vector machines. *IEEE Geoscience and Remote Sensing Letters*, 4(4):674–677, 2007.

[11] Wei Hu, Yangyu Huang, Li Wei, Fan Zhang, and Hengchao Li. Deep convolutional neural networks for hyperspectral image classification. *Journal of Sensors*, 2015:1–12, 07 2015.

[12] Spectral Imaging Ltd. Specim. Specim FX17. https://www.specim.fi/products/specim-fx17. Abgerufen am 17.12.2021.

[13] Spectral Imaging Ltd. Specim. Specim FX17 - Datasheet. https://www.specim.fi/wp-content/uploads/2020/03/Specim-FX17-Technical-Datasheet-02.pdf. Abgerufen am 17.12.2021.

[14] MTD Gmbh. MTD Linienleuchten. http://mtd-gmbh.de/pdfs/MTD-LED-2022.pdf. Abgerufen am 20.12.2021.

[15] Aurélien Géron. *Praxiseinstieg Machine Learning mit Scikit-Learn, Keras und TensorFlow, 2nd Edition.* dpunkt, 2nd edition edition, 2020. 852.

[16] Stuart J. Russell, Peter Norvig, Frank Kirchner, and Frank Langenau. *Künstliche Intelligenz : ein moderner Ansatz.* it - Informatik. Pearson, Higher Education, München, 3., aktualisierte auflage edition, [2012]2012. 1307 Seiten ; 25 cm : Illustrationen, Diagramme.

[17] Martin Werner. *Neuronale Netze mit Faltungsschichten*, pages 409–465. Springer Fachmedien Wiesbaden, Wiesbaden, 2021.

[18] Wolfgang Ertel. *Neuronale Netze*, pages 285–349. Springer Fachmedien Wiesbaden, Wiesbaden, 2021.

[19] Martin Werner. *Flache Neuronale Netze für die Klassifizierung*, pages 349–382. Springer Fachmedien Wiesbaden, Wiesbaden, 2021.

[20] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights Imaging*, 9:611–629, 2018.

[21] cs231n. Convolutional neural networks. https://cs231n.github.io/convolutional-networks/.

[22] Arc. Convolutional neural network. https://towardsdatascience.com/convolutional-neural-network-17fb77e76c05, 12 2018.

[23] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *J. Mach. Learn. Res.*, 11:625–660, March 2010.

[24] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3320–3328, Cambridge, MA, USA, 2014. MIT Press.

[25] Shruti Jadon. A survey of loss functions for semantic segmentation, 10 2020.

[26] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[27] Phil Wennker. *Machine Learning*, pages 9–37. Springer Fachmedien Wiesbaden, Wiesbaden, 2020.

[28] Martin Werner. *Künstliche Neuronen und Lernen*, pages 319–347. Springer Fachmedien Wiesbaden, Wiesbaden, 2021.

[29] Martin Werner. *Lernen mit dem Backpropagation-Algorithmus*, pages 383–408. Springer Fachmedien Wiesbaden, Wiesbaden, 2021.

[30] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[31] Wolfgang Ertel. *Maschinelles Lernen und Data Mining*, pages 201–283. Springer Fachmedien Wiesbaden, Wiesbaden, 2021.

[32] Claude Sammut and Geoffrey I. Webb, editors. *Accuracy*, pages 8–8. Springer US, Boston, MA, 2017.

[33] Claude Sammut and Geoffrey I. Webb, editors. *F1-Measure*, pages 497–497. Springer US, Boston, MA, 2017.

[34] Reuben R Shamir, Yuval Duchin, Jinyoung Kim, Guillermo Sapiro, and Noam Harel. Continuous dice coefficient: a method for evaluating probabilistic segmentations. *bioRxiv*, 2018.

[35] Kai Ming Ting. *Confusion Matrix*, pages 260–260. Springer US, Boston, MA, 2017.

[36] Christoph Molnar. *Interpretable Machine Learning*. 2 edition, 2022.

[37] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization.