Christian-Albrechts-Universität zu Kiel
Institut für Informatik

Bachelorarbeit

Verwendung von Image Embedding Ähnlichkeitsmessungen zum Clustering von Fixed Gear Schiffstrajektorien

Bahne Thiel-Peters
Matrikel Nr. 1159242

Betreuer:
Prof. Dr. Peer Kröger,
Mirjam Bayer

Christian-Albrechts-Universität zu Kiel
Institut für Informatik
Informationssysteme und Data Mining
Olshausenstr. 40, 24098 Kiel

29. September 2023

Kurzfassung

Diese Arbeit beschäftigt sich damit, die verschiedenen Unterarten des Fixed Gear Fischens unterscheiden und erkennen zu können. Dies beinhaltet die Auswahl einer Metrik, welche möglichst viele vorhandene Informationen aus Trajektorien der Schiffe benutzt und abbildet, um den Schiffen Distanzwerte zuzuordnen. Die implementierte Metrik basiert dabei auf einem trainierten ResNet50 Model, welches visuelle Darstellungen der Schiffstrajektorien als Inputparameter erhält und diese Bilder der Routen mit Features einbettet. Basis dieser Arbeit bietet ein von Global Fishing Watch zur Verfügung gestellter Datensatz, der eine nach Schiffen sortierte Sammlung von AIS Nachrichten enthält. Die Metrik wird durch den Vergleich von Streudiagrammen und dem Silhouettenkoeffizienten bewertet.

- Trajektorie
- Metrik
- Neuronales Netzwerk
- ResNet50
- Transfer Learning
- Image Embedding
- Feature Vector
- Clustering
- Fixed Gear

Abstract

This work focuses on distinguishing and recognizing the various subtypes of Fixed Gear fishing. This involves selecting a metric that utilizes and maps as much available information as possible from ship trajectories to assign distance values to the ships. The implemented metric is based on a trained ResNet50 model, which takes visual representations of ship trajectories as input parameters and embeds these images of routes with features. The basis of this work is a dataset provided by Global Fishing Watch, which contains a collection of AIS messages sorted by ships. The metric is evaluated through the comparison of scatter plots and the silhouette coefficient.

- trajectory
- metric
- neural network
- ResNet50
- transfer learning
- image embedding
- feature vector
- clustering
- fixed gear

Inhaltsverzeichnis

Kurzfassung II

Inhaltsverzeichnis IV

1 Einleitung 1

2 Grundlagen 2
2.1 Automatic Identification System (AIS) . . . 2
2.2 Datensatz . . . 3
2.3 Fischfangtechniken . . . 3
2.3.1 Fixed Gear . . . 3
2.3.2 Trawler . . . 4
2.4 Metriken . . . 4
2.5 Clustering . . . 5
2.5.1 Principal Component Analysis (PCA) . . . 5
2.5.2 Silhouettenkoeffizient . . . 6
2.6 Feature Vector (FV) . . . 6

3 Stand der Technik 8
3.1 Metriken für Zeitreihen . . . 8
3.1.1 Dynamic Time Warping (DTW) . . . 8
3.1.2 Longest Common Subsequence (LCS) . . . 10
3.2 Image Embedding . . . 11

4 Implementation 13
4.1 Abbildung von Trajektorien auf ein Bild . . . 13
4.1.1 Filtering . . . 13
4.1.2 Grundkonstrukt . . . 15
4.1.3 Skalierung . . . 16
4.1.4 Geschwindigkeit . . . 17
4.1.5 Wassertiefe . . . 18
4.1.6 Alle Features . . . 20
4.1.7 Trawlers . . . 20
4.1.8 Ausführung . . . 22
4.2 Fine Tuning . . . 22
4.3 Metrik . . . 23
4.4 Clustering . . . 23
4.4.1 Umsetzung . . . 23
4.4.2 Ausführung . . . 23

5 Evaluation **24**
5.1 Auswertung 24
5.1.1 Grundkonstrukt 24
5.1.2 Skalierung 25
5.1.3 Geschwindigkeit 26
5.1.4 Wassertiefe 27
5.1.5 Alle Features 29
5.2 Fazit und Ausblick 31

Literatur **32**

Abbildungsverzeichnis **34**

Tabellenverzeichnis **35**

Erklärung **36**

1 Einleitung

Illegales Fischen und die weltweite Überfischung der Meere sind Themen wachsender globaler Bedeutung. Das Ausmaß der Fischaktivitäten bedroht die Artenvielfalt und bringt eine Menge wirtschaftlicher und sozialer Probleme mit sich. Ende 2021 verbot Deutschland das Nutzen von Stellnetzen in manchen Schutzgebieten der Ostsee, um dadurch entstehenden Beifang zu reduzieren. Immer mehr an Wasser angrenzende Länder bewegen sich langsam in diese Richtung und schließen sich Deutschland an. Um das durch das wachsende Verbot sinkende Angebot wieder zu steigern, steigt die Anzahl sogenannter *Dark Vessels*. Dark Vessels sind Schiffe, welche unangemeldet und illegal die Gesetzeslage durch Ausschalten des verpflichtenden Identifikationssystems AIS 2.1 oder unter auftreten einer falschen Identität ignorieren.

Moderne Satellitenüberwachung bietet mit neuen Technologien die Möglichkeit Daten über solche Dark Vessels anzusammeln, was einen wichtigen Schritt in der Bekämpfung des illegalen Fischfangs darstellt. Die Nutzung dieser Daten ist essenziell für die Überwachung von Schiffen und ermitteln der von diesen genutzten Fischfangtechnik. Die Vorhandene Forschungsliteratur enthält zahlreiche Arbeiten mit dem Ziel, Fischfangtechniken, wie Trawler, Pure Seiners, Troller und Fixed Gear klassifizieren zu können. Allerdings fehlt es an Arbeiten, welche die unterschiedlichen Fischfangtechniken innerhalb der Fixed Gear Klasse klassifizieren.

Die vorliegende Arbeit thematisiert das Erkennen und Identifzieren der unterschiedlichen Fixed Gear Fischfangtechniken durch das Einsetzen einer Metrik zur Berechnung einer Distanzmatrix. Das erste Kapitel erläutert die Grundlagen dieser Arbeit. Im Stand der Technik wird ein Blick auf die bisher entwickelten Techniken zum Lösen ähnlicher Probleme geworfen, sowie die Auswahl der in dieser Arbeit zu entwickelnden Metrik getroffen. Anschließend erfolgt die Implementation, in der die theoretisch erläuterten Konzepte praktisch umgesetzt werden, wodurch ein Programm entsteht, welches die einzelnen Schiffe in Klassen einordnet. Zuletzt wird die Implementation ausgewertet.

2 Grundlagen

2.1 Automatic Identification System (AIS)

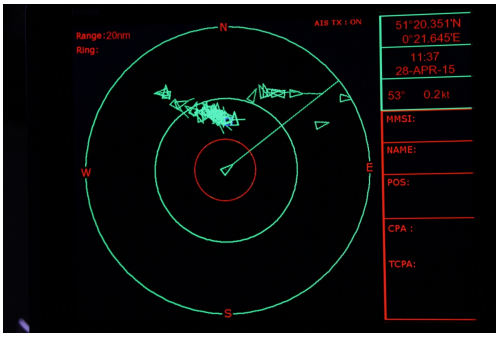


Abbildung 2.1: Bildschirm eines AIS Radars, [Vic20]

Der Begriff Automatic Identification System beschreibt ein automatisches, autonomes und funkgestütztes Lokalisierungssystem, welches von Schiffen genutzt wird. Schiffe versenden und empfangen AIS Nachrichten über einen Transceiver und tauschen somit Daten über das Verhalten des Schiffes untereinander aus. Wichtiger Bestandteil der AIS Anlage ist ein Bildschirm wie in Abb. 2.1, welcher die eigene Position und die der Schiffe in der Nähe angibt. Die Nutzung eines solchen AIS Geräts ist seit Dezember 2004 für alle Schiffe mit mindestens 300 BRZ[1] vorgeschrieben [Mar21] und somit aktueller Standart. Gründe der Nutzung von AIS sind hauptsächlich folgende:

- Kollisionsverhütung zwischen Schiffen
- Informationserhalt von Küstenstaaten
- Lekung des Schiffverkehrs
- Überwachung des illegalen Fischfangs.

In AIS Nachrichten enthaltene Daten sind unter anderem Statische Schiffsdaten, wie die MMSI[2] , Schiffsname, sowie Schiffstyp, Dynamische Schiffsdaten wie GPS Koordinaten, Richtung, Geschwindigkeit und Reisedaten, wie zum Beispiel die Gefahrgutklasse der Ladung, das Ziel des Schiffes, und die Passagieranzahl bei Passagierschiffen [Mar20].

[1] BRZ steht für Bruttoraumzahl und ist eine Art die Größe eines Schiffes anzugeben.
[2] MMSI, Maritime Mobile Service Identity, Eine einzigartige neunstellige Rufnummer, welche ein Schiff identifiziert.

2.2 Datensatz

mmsi	timestamp	distance_from_shore	distance_from_port	speed	course	lat	lon	is_fishing	source
7572518792420	1347663871	0	36054.625	0	0	42.7987480164	-8.9449920654	-1	gfw
7572518792420	1348056426	0	36054.625	0	0	42.7987174988	-8.9450750351	-1	gfw
7572518792420	1350409469	0	90970.296875	0	198.1999969482	43.1064186096	-9.2154664993	-1	gfw

Tabelle 2.1: Ein Ausschnitt aus dem Datensatz

Der verwendete Datensatz basiert auf einem Datensatz von Global Fishing Watch und enthält ausschließlich Daten von Schiffen, welche Fixed Gear fishing betreiben. Tabelle 2.1 zeigt einen Ausschnitt aus dem Datensatz, welcher sich aus einer Reihe an AIS Nachrichten zusammen setzt, welche jeweils aus zehn verschiedenen Daten bestehen. Unter anderem sind das die MMSI eines Schiffes, welche einzigartig ist und so direkt einem Schiff zugewiesen werden kann, und die GPS Position. Alle AIS Nachrichten, also Zeilen im Datensatz, eines Schiffes betrachtet nennt man Trajektorie und kann auch als Route betrachtet werden, welche ein Schiff fährt. Neben den durch die AIS Nachrichten vorhandenen Daten, werden weitere Daten durch die Arbeit einer anderen Arbeit des selben Instituts zur Verfügung gestellt [BFDK23]. Das sind zum einen die Wassertiefe und zum anderen ein Trip count, welcher allen AIS Nachrichten, welche zum selben Trip gehören die selbe Zahl zuweist. Ein Trip ist hier als die Menge an AIS Nachrichten, welche in die Zeitspanne des Ablegens bis zum nächsten Anlegensfallen. Neben den weiteren Daten in den Trajektorien, wurden durch die genannte Arbeit die Daten gesäubert, und die Anzahl der Ausreißer in den Daten wurden minimiert, wodurch Ausgangslage der vorliegenden Arbeit reine Daten sind.

Die Datensammlung erfolgte von Januar 2012 bis November 2016 durch die Dalhousie University, Global Fishing Watch selbst, so wie einer Crowdsourcing Kampagne. Insgesamt enthält der Datensatz Daten von 36 verschiedenen Schiffen, wobei die kleinste Trajektorie aus 6.502 und die größte aus 67.040 AIS Nachrichten besteht.

2.3 Fischfangtechniken

2.3.1 Fixed Gear

Der Begriff Fixed Gear ist ein Oberbegriff von Fischfangtechniken, welche gemeinsam haben, dass die Ausrüstung an Positionen aufgestellt wird und anders als bei anderen Techniken, wie Trawlers, nicht am Schiff befestigt ist. Schiffe fahren auf das Meer hinaus, fixieren die Ausrüstung an verschiedenen Stellen, und sammeln es nach unterschiedlichen Zeiten wieder samt gefangenen Tieren ein. Typische Unterarten sind zum einen *stationary longlines:* Leinen mit vielen Köderhaken, welche benutzt werden, um Kabeljau, Schwertfische, Haie und weitere zu fangen. Potting/Trapping beschreibt die Anwendung von aus Holz, Plastik oder Seilen bestehenden Käfigen, welche Hummer und Krabben fangen. Die letzte übliche Fixed Gear Art ist das Gillnetting, welches im Wasser aufgestellte Netze

sind, in welchen sich Fische verfangen und verheddern. Typische Zielfische sind hier zum Beispiel Heringe, Schollen und Forellen.

Da die unterschiedlichen Fixed Gear Unterarten unterschiedliche Fische fangen, ist auch mit unterschiedlichen Ausprägungen von Eigenschaften zu rechnen. So könnte es sein, dass die Langleinen mit größerer Wahrscheinlichkeit in hoher Wassertiefe und größerer Küstendistanz stehen, als Traps. Durch die höhere Distanz würde das Schiff ebentuell öfter zum Hafen zurückkehren. Die Größe der Schiffe kann eine Rolle bei der Identifizierung der Unterart spielen, falls Schiffe welche Trapping und Potting betreiben mehr Stops machen, als die anderen. Ebenfalls die Zeit, welche durch das Aufstellen der Ausrüstungsgegenstände benötigt wird kann eine Rolle spielen. So benötigt das Aufstellen eines Krabbenkäfigs vermutlich weniger Zeit, als das Aufstellen einer fünf Kilometer langen Langleine.

2.3.2 Trawler

Trawler sind eine eigene Klasse von Fischfangtechniken. Hier wird, anders als beim Fixed Gear, die Ausrüstung am Schiff montiert. Die Ausrüstung sind hier Schleppnetze, welche an der Seite eines Schiffes montiert ist. Während des Fischens wird das Schleppnetz in Bodentiefe in das Wasser gelassen und Fische werden durch das Fahren des Schiffes in das Netz gedrängt. Trawler fahren typischer Weise hinaus, bis sie zur gewünschten Tiefe kommen, und fahren dort eine längere Zeit in einer langsamen Geschwindigkeit.

2.4 Metriken

Die Distanz zwischen zwei Objekten q,p, wie Mengen oder natürlichen Zahlen, ist nicht einheitlich definiert, weshalb es verschiedene Möglichkeiten gibt, diese zu berechnen. Metriken oder auch Distanz Funktionen sind Funktionen, welche die Distanz zwischen zwei Mengen berechnen. Eine weit verbreitete Metrik ist die euklidische Metrik, welche wie folgt definiert ist:

$$d(q,p) = \sqrt{\sum_{j=1}^{n} (q_j - p_j)^2}$$

Diese berechnet die länge der Linie, welche die beiden Punkte q und p verbindet. Ein Spezialfall der Metrik ist der zwei dimensionale Raum, da die Berechnung der Metrik dort dem Satz des Pytagoras entspricht. Sind die beiden Mengen gleich, erhalten sie einen Distanzwert von 0. Bei steigender Abweichung einander nach Art der Metrik, steigt ihr Distanzwert. Die resultierenden Distanzmatrixen bieten den Grundbaustein zur Anwendung von Clusteringalgorithmen wie K-means.

2.5 Clustering

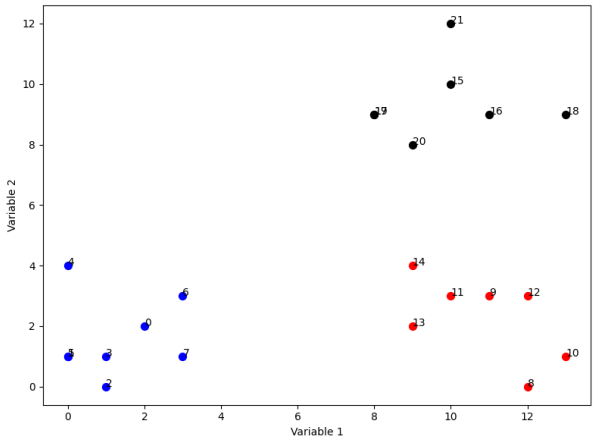


Abbildung 2.2: Streudiagramm eines Beispielclusterings, [kme23]

Clustering ist eine Methode des unsupervised machine learnings, welche eine Menge von Datenpunkten in Clusters oder auch Klassen einteilt. Vorteil dieses Vorgehens ist, dass der Clusteringalgorithmus keine Vorkenntnisse über die Klassen der Datenpunkte benötigt, sondern die Datenpunkte nur auf Grundlage von Mustern und Ähnlichkeit in den Daten selbst in Klassen einteilt. Die Informationen welche über die Datenpunkte zur Verfügung stehen, werden im Streudiagramm in Abb. 2.2 auf den Achsen eingetragen und die durch den Algorithmus zugeteilte Klasse ist durch die Farbe des Punktes zu erkennen. Das Ziel des Clusterings ist eng beieinander liegende Punkte in eine Klasse einzuteilen, wobei die Klassen idealerweise weit auseinander liegen. Die Anzahl der Klassen wird je nach Art des Clusterings vor dem Durchlauf festgelegt, oder währenddessen bestimmt. K-means ist ein *Partitional Clustering*, was bedeutet, dass die Anzahl der Klassen k vor der Ausführung festgelegt werden [Tre16]. Dies ist im Falle dieser Arbeit möglich, da die Klassen die verschiedenen Fixed Gear Kategorien darstellen.

2.5.1 Principal Component Analysis (PCA)

Inputparameter eines Clusterings ist oftmals und auch in diesem Fall eine Distanzmatrix. Da diese Matrix den Abstand aller n Schiffe zueinander enthält, findet das Clustering als solches auch im Raum $\mathbf{R}^n$ statt. Um die Datenpunkte in einem Streudiagramm im zweidimensionalen Raum $\mathbf{R}^2$ abzubilden, findet eine Reduzierung der n Dimensionen auf 2 statt. Für eine solche Reduzierung wird PCA herangezogen. Hier wird nach Korrelationen in den Daten gesucht, sodass mehrere Daten in einer Variable, einem *Component* festgehalten werden. Da eine solche Reduzierung immer einen Informationsverlust impliziert, ist die Darstellung des Streudiagramms nur eine Annäherung der tatsächlichen Darstellung.

2.5.2 Silhouettenkoeffizient

Um das Clustering zu bewerten können verschiedene koeffizienten herangezogen werden. Der meistgenutzte ist hierbei der Silhouettenkoeffizient S_C. Dieser gibt Auskunft darüber, wie eng die Punkte innerhalb der Klassen beieinander liegen und wie weit sie von den anderen Klassen entfernt sind. Der Koeffizient wird für den Punkt i wie folgt definiert:

$$s_i = \frac{b_i - a_i}{max(a_i, b_i)}$$

wobei a_i der durchschnittliche Abstand von i zu allen anderen Punkten in der selben Klasse, und b_i der durchschnittliche Abstand von i zur nächstgelegenen Klasse [R22] ist. Dabei wird der Abstand zwischen den beiden Punkten q und p hier definiert als der euklidische Abstand d im n-dimensionalen Raum $\mathbf{R}^n$ 2.4. Um nun den Silhouettenkoeffizienten des ganzen Clusterings zu berechnen, wird der Durchschnitt aller Koeffizienten der einzelnen Punkte berechnet. Durch die Art der Berechnung gilt für den Koeffizienten $s \in [-1, 1]$, wobei -1 für gar keine und 1 für vollkommene Isolation der Klassen ineinander steht. Der S_C des Diagramms aus Abb. 2.2 beträgt 0.6731.

2.6 Feature Vector (FV)

Ein *Feature Vector* ist eine numerische Repräsentation eines Bildes. Diese enthält eine große Anzahl an Werten, welche verschiedene Dinge über das Bild aussagen.

Abbildung 2.3: Beispiel von Image Embedding, [Med23]

Abb. 2.3 zeigt einen Vergleich zwischen zwei verschiedenen Eiswaffeln, wobei im unteren Teil des Bildes der *Feature Vector* eingezeichnet ist. Um ein Bild in eine solche numerische Representation zu verwandeln, benötigt es ein darauf trainiertes Neuronales Netzwerk. Die Daten zum Trainieren des Modells stammen aus dem sogenannten ImageNet Datensatz, was eine riesige strukturierte Datenbank von Bildern ist. Sie enthält über 14 Millionen Bilder, zu welchen zusätzlich die Kategorie des Bildes festgehalten wird.

Ein Modell zur Bilderkennung besteht aus vielen verschiedenen Schichten, welche durchlaufen werden. Diese verschiedenen Schichten wenden verschiedene Operationen

auf das Bild an, um Muster und Merkmale zu erkennen. Die Feature Schicht wertet die vorherigen aus, und hält diese Auswertungen in einer hochdimensionalen Liste, dem Feature Vector, fest [Mok21]. Die einzelnen Dimensionen des Feature Vectors enthalten zahlreiche Informationen wie Texturen, Kantenglätten, Kontraste und Farben, können sich aber auch auf nur kleine Teile des Bildes beschränken. Da ein solcher Vektor aus mehreren tausend Dimensionen bestehen kann, enthält er semantische Informationen, also Muster und Strukturen über das Bild.

3 Stand der Technik

3.1 Metriken für Zeitreihen

Stehen Informationen in Verbindung mit Zeitpunkten, so spricht man von Zeitreihen/time series. Im Falle von Trajektorien können diese Informationen aus den Koordinaten, oder dem Abstand zur ersten Koordinate sein. So könnte die jeweilige Zeitreihe pro Schiff aus der Distanz zum geographischen Punkt am ersten Zeitpunkt bestehen. Um nun einen Vergleich zwischen diesen Zeitreihen zu ziehen, muss eine geeignete Metrik her. Die euklidische Metrik lässt sich auch als Metrik für Zeitreihen einsetzen, indem eine Dimension der Zeit entspricht. Problem dieser Distanzberechnung ist, dass die gleiche Anzahl an Zeitpunkten in den beiden Zeitreihen vorausgesetzt wird. Es gibt zahlreiche weitere Metriken für Zeitreihen, Dynamic Time Warping und Longest Common Subsequence sind welche, die oft in Zusammenhang mit Vessel Clustering benutzt werden. Folgend werden beide Metriken erläutert und Probleme dargelegt, welche im Rahmen dieser Arbeit bestehen.

3.1.1 Dynamic Time Warping (DTW)

Dynamic Time Warping wurde ursprünglich entwickelt um verschiedene Sprachmuster unterscheiden zu können. Die Mengen bestehen in diesem Fall also aus den Datenpunkten einer Audioaufnahme [Verweis]. Nimmt eine Person einen Text auf zwei verschiedene Weisen, schnell und langsam, auf, so berechnet die Metrik eine sehr geringe Distanz, da bis auf die Verzerrung in der Zeit kaum ein Unterschied in der Amplitude besteht.

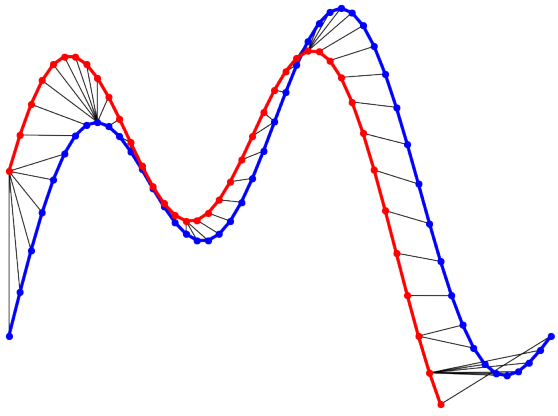


Abbildung 3.1: Beispiel von Dynamic Time Warping, [Ali23]

Die Abbildung 3.1 zeigt zwei Mengen, in rot und blau, wobei auf der X-Achse die Zeit und auf der Y-Achse ein Feature steht. Im Beispiel der Spracherkennung steht auf der Y-Achse die Amplitude, beziehungsweise die Lautstärke. Anders als auf der Abbildung, ist es aber auch möglich mehrere Features im Verlauf der Zeit darzustellen, was ein Argument für die Nutzung dieser Metrik im Vergleich von Schiffstrajektorien ist.

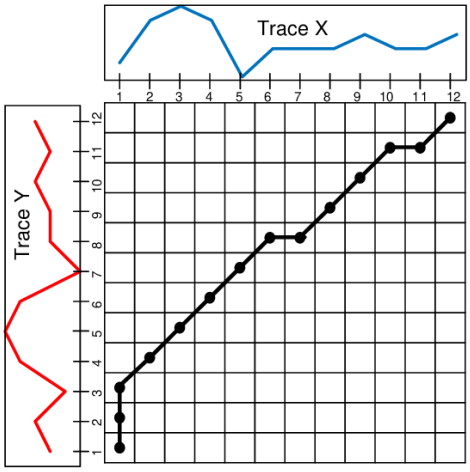


Abbildung 3.2: Die Berechnungsmatrix von DTW mit eingezeichnetem Pfad, [GDD+19]

Der Algorithmus zur Berechnung der durch DTW entstehenden Distanzmatrix M erlaubt, anders als die euklidische Metrik, einen Unterschied in den Längen der zu vergleichenden Mengen X und Y. Das Vorgehen ist wie folgt: Berechne zwischen allen Elementen $x \in X$, $y \in Y$ an den Zeitpunkten $i, j \in \mathbf{R}$ den Abstand d:

$$d(x,y) = |x[i] - y[j]| + \min \begin{pmatrix} M[i-1,j] \\ M[i,j-1] \\ M[i-1,j-1] \end{pmatrix}$$

Anhand der Matrix M kann nun die Distanz zwischen X und Y errechnet werden. Dafür wird wie in Abb. 3.2 ein Pfad bestimmt, welcher im Element oben rechts startet, und bis zum Element unten links geht. Dazwischen wird jeweils das der drei Elemente $M[i-1,j], M[i,j-1], M[i-1,j-1]$ dem Pfad hinzugefügt, welches den kleinsten Wert hat. Die Distanz ist nun die Summer aller Elemente des Pfads.

Im Kontext dieser Arbeit könnten die Features in Abhängigkeit der Zeit die Koordinaten, und die Wassertiefe sein. Ähnlich wurde DTW bereits im Kontext von Erkennung von Unterschieden bei Wasserstraßen und Flüssen verwendet [ZS18]. In diesem Fall berechnet DTW Schiffen, welche in unterschiedlicher Geschwindigkeit durch den selben Fluss fahren, eine geringe Distanz. Wird in gleicher Geschwindigkeit durch verschiedene Flüsse gefahren, so ist der Distanzwert größer. Für Trajektorien auf dem offenen Meer wird ein geringerer Distanzwert ermittelt, wenn sich die Struktur der Routen ähnelt. Gibt es Unterschiede in der

Route zwischen Hafen und den ersten Ausrüstungsorten, so wird dies von DTW womöglich erkannt, denn eine längere Anreise wird einer kürzeren zugeordnet. Allerdings wird es bei Schiffen die in der selben Klasse, jedoch unterschiedlich groß sind zu Unterschieden in der Anzahl der Ausrüstungsorten kommen. Das zeichnet sich dann dadurch aus, dass ein Schiff mehr Stopps hinlegt, was graphisch bei DTW durch einen längeren Graphen sichtbar wäre. Zusätzlich gibt es das Problem, dass die Art der Berechnung von dem Fakt der Wassertiefe beeinflusst wird, wodurch die Wassertiefe berücksichtigt wird, welche gar nicht an Orten des Fischens liegt.

3.1.2 Longest Common Subsequence (LCS)

Um verschiedene Sequenzen von Zeichen miteinander vergleichen zu können, findet die Longest Common Subsequence Anwendung. Damit wird beispielsweise der Werdegang von Sprachen in der Linguistik, oder auch DNA in der Bioinformatik analysiert. Dabei wird bei zwei Mengen X,Y von Zeichen der längste Teilsequenz gesucht, also die Zeichenkette von Zeichen, welche in beiden Mengen vorkommt.

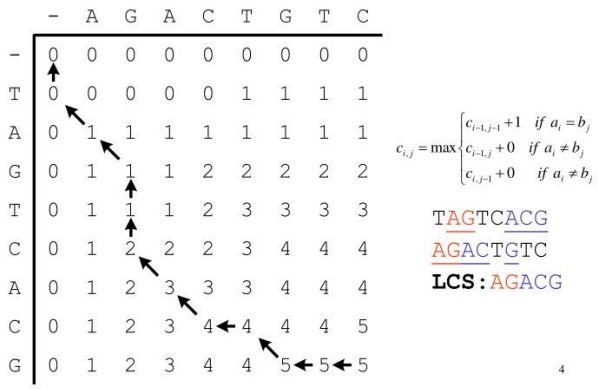


Abbildung 3.3: Die Berechnungsmatrix von LCS, [don14]

Die Berechnung der LCS funktioniert dabei auch über eine Berechnungsmatrix M, welche zweidimensional in Größe der Längen der Sequenzen ist. Bei der Initialisierung der Matrix, iteriert der Algorithmus dabei verschachtelt über beide Sequenzen, wobei dabei zwei Zeiger $i, j \in \mathbf{R}$ existieren, welche beide für eine der beiden Sequenzen stehen. Dabei zeigen die Zeiger immer auf ein Zeichen der Sequenz, und stimmen die aktuellen Zeichen überein, so wird der Wert so wie in Abb. 3.3 fetgelegt. Ähnlich wie bei DTW existiert wieder ein Pfad, wobei dieser Pfad gibt die längste Teilsequenz angibt. Erstellt wird er wieder durch das gehen vom letzten Element der Matrix bis zum ersten, aber diesmal wird immer zu dem Element gegangen, von dem das aktuelle Element die Erhöhung des Wertes hat. Bevor eine Schräge gegangen wird, was der Fall ist wenn die aktuellen Zeichen i,j übereinstimmen, kann i am Ende der Ergebnissequenz notiert werden.

Dass LCS auch in einem ähnlichen Kontext einer Arbeit funktioniert, wurde bereits in [MT09] oder [BTTT18] gezeigt. Um LCS als eine passende Metrik für diese Arbeit zu

implementieren, müssen die AIS-Nachrichten mit PCA auf einen Wert beschränkt werden, da LCS nur eindimensionale Sequenzen erlaubt. Außerdem muss eine Relativierung der Koordinaten stattfinden, sodass es nicht die Koordinaten sind die übereinstimmen müssen, sondern die Distanz zum ersten Punkt. Das bei DTW auftretende Problem der unterschiedlichen Anzahl an Ausrüstungsorten tritt hier nur in einem geringeren Maße auf, da LCS wie in Abb. 3.3 nach keiner passenden Übereinstimmung trotzdem weiterrechnet. Problematisch ist zudem die Distanz welche von Schiffen zurückgelegt wird. Fischen zwei Schiffe in der selben FixedGear Klasse und somit in einem ähnlichen Muster, allerdings legt ein Schiff eine höhere Distanz zurück, so findet die Berechnung der LCS keine Übereinstimmung mehr und auch die danach auftretenden Datenpunkte werden eventuell nicht mehr der längsten Teilsequenz zugeordnet.

3.2 Image Embedding

Eine dritter Ansatz ergibt sich aus dem Nutzen eines Image Embedders. Ein Image Embedder ist ein Neuronales Netzwerk, welches einem Bild einen Feature Vector 2.6, also eine numerische Repräsentation, zuordnet und dieses mit diesem einbettet. Eine Metrik macht sich diese Einbettung zunutzen und berechnet den Abstand zweier Bilder zum Beispiel mit der Cosine Similarity.

Anhand der *Cosine Similarity* C_S ist es möglich zwei Vektoren X und Y zu vergleichen, in dem der Kosinus des Winkel zwischen ihnen ausgerechnet wird.

$$C_S(X,Y) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2 \sum_{i=1}^{n} B_i^2}}$$

Dabei stehen A_i, B_i für das i-te Element von A und B und n ist die Länge [Pra18]. Für C_S gilt dabei immer $C_S \in [-1,1]$, wobei für $C_S = 1$ gilt, dass die Vektoren übereinstimmen, also in die selbe Richtung zeigen und für $C_S = 1$, dass beide in die entgegengesetzte Richtung zeigen, also das Gegenteil voneinander sind.

Um die Einbettung der Bilder in Vektoren als Metrik zur Berechnung der Distanzmatrix von Schiffstrajektorien zu benutzen, müssen zunächst Bilder der einzelnen Trajektorien gezeichnet werden. Eine Möglichkeit ist das zeichnen der Route, die ein Schiff gefahren ist, in dem Linien zwischen den Punkten gezeichnet werden, welche im Datenset als AIS-Nachrichten vorliegen. Zusätzlich können weitere Informationen wie die Geschwindigkeit, die Wassertiefe und die Größe der Trajektorie berechnet und in das Bild kodiert werden. Diese Informationen bieten dem ImageEmbedder eine Grundlage semantische Informationen wie Muster aus dem Bild zu extrahieren und dadurch die Ähnlichkeit zwischen Fixed Gear Klassen zu erkennen. Anders als in 3.1.1 und 3.1.2 kann hier eine Vielzahl von Features berücksichtigt und gegebenenfalls leicht angepasst werden. Zusätzlich könnte das Modell, anders als die vorherigen, aufgrund der großen Anzahl an Schichten stabiler gegenüber einer variierenden Anzahl an Ausrüstungsorten sein, da es das Routenmuster womöglich trotzdessen erkennen kann. Das bedeutet, dass zwei Trajektorien mit der gleichen Technik,

welche sich aber hinsichtlich der Nachrichtenanzahl unterscheiden, eine geringe Distanz zueinander erhalten. Ein geeignetes CNN ist das ResNet50 Modell. Dieses wurde bereits erfolgreich zur Erkennung und Klassifizierung von Handschriften genutzt [GAKJ22], was auch eine semantische Mustererkennung voraussetzt. Das Modell besteht aus 50 Schichten, wodurch komplexe Mustererkennung ermöglicht wird. Um das gewählte Modell gezielt für diese Arbeit zu nutzen, ist es möglich *Transfer Learning* mit dem Modell zu betreiben. Transfer Learning trainiert das Modell auf aufgabenspezifische Objekte, in diesem Fall Bilder der Trajektorien. Um das umzusetzen, wird eine neues Modell erstellt, welches dem bisherigen ResNet50 Modell ohne der Ausgangsschicht entspricht. Die Ausgangsschicht enthält die Vorhersagen des Bildes, also Werte aller Klassen. Da diese Arbeit nicht mit den Vorhersagen der im ResNet50 Modell enthaltenden Klassen arbeitet, kann diese Schicht entfernt werden. Nötig ist nun eine Schicht, welche die Features der Bilder erkennt und in einen FV extrahiert. Typisch für diese Aufgabe ist das Nutzen einer *GlobalAveragePooling2D (GAP)* Schicht. Features der Bilder sind in sogenannten *Feature Maps* enthalten, welche unterschiedliche Filter auf die Bilder anwenden und unterschiedliche Merkmale erkennen. GAP berechnet nun den Durchschnitt aller *Feature Maps* und fügt diese nach und nach dem FV hinzu [Mwi22]. Das Training zielt dabei darauf ab, nur diese oberste GAP Schicht zu trainieren, und die anderen Schichten einzufrieren.

4 Implementation

Die theoretisch beleuchtete Idee der Implementation gilt nun praktisch durchzuführen. Für die Implementierung wird Python in der Version 3.8.10 herangezogen.

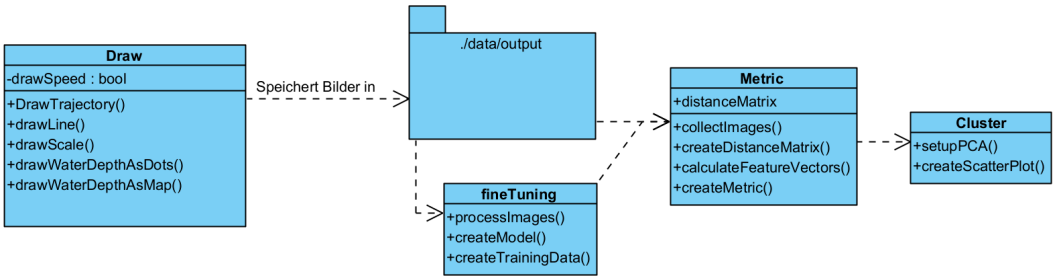


Abbildung 4.1: Klassendiagramm der Klassen der Implementation

Die Implementation ist aufgeteilt in vier Schritte, welche alle jeweils in einer Klasse implementiert wurden: das Darstellen einer Trajektorie als ein Bild, das fine tuning des ResNet50 Modells, das anschließende Erstellen einer Distanzmatrix, sowie die Kategorisierung durch ein Clustering. Dieses Kapitel beschreibt die Idee hinter der Umsetzung der jeweiligen Klasse, sowie eine Dokumentation, welche das Aufrufen sowie die verschiedenen Einstellungen erläutert. Das Repository dieser Arbeit kann unter (https://git.informatik.uni-kiel.de/stu229354/bahne-tp-bachelor-thesis) aufgerufen werden.

4.1 Abbildung von Trajektorien auf ein Bild

Die Grundidee ist, jeden verfügbaren Datensatz eines Schiffes als ein eigenes Bild darzustellen. Diese Art der Implementation erlaubt es, mehrere Features zu visualisieren, welche verschiedene Informationen über die Trajektorie geben. Wie Informationen in das Bild kodiert werden, kann dabei auf verschiedene Möglichkeiten passieren. Ziel des Ansatzes ist, dem Feature Embedding einen großen Informationsgehalt über die Bilder zu geben, und somit die Schiffe besser clustern zu können.

Die Implementierung nutzt die Python Bibliothek Pillow, welche verschiedene Operationen auf Bildern anbietet.

4.1.1 Filtering

Noch bevor die Daten visualisiert werden können, müssen die Daten durch ein Filtering ausgewählt werden. Ziel hierbei ist, jedem Schiff eine Trajektorie zu geben, welche die

Route des Schiffes möglichst fehlerfrei und genau angibt. Hierfür wurden vier verschiedene Ansätze verfolgt, wobei hier der Verlauf des Filterings dargestellt ist, und Vor- und Nachteile aufgezeigt werden.

- **Filtern der Daten nach Zeit:** Jedes Schiff erhält als Trajektorie alle Daten aus dem ersten vorliegenden Monat. Dieser Ansatz verfolgt die Idee, einen Kontrast zwischen Fangmethoden darzustellen welche unterschiedlich viel fischen. Da die Frequenz der zu sendenden AIS-Nachrichten von verschiedenen Faktoren abhängt, senden manche Schiffe innerhalb eines Zeitraumes ein Vielfaches von dem was ein anderes Schiff sendet. Zudem kommen Einzelfälle von Schiffen, welche innerhalb eines Monats nur wenige bis keine Daten senden. Eine zu große Diskrepanz der Nachrichtenanzahl zwischen den Trajektorien verfälscht das Ergebnis des Image Embeddings legt den Fokus dabei zu sehr auf die Anzahl der Nachrichten.

- **Filtern nach der Anzahl der Trajektorien:** Die Trajektorie wird nun gleich den ersten n AIS-Nachrichten gesetzt, wobei $n \in \mathbf{N}$ gilt. Hierdurch wird eine einheitliche Anzahl garantiert, was die Problematik der fast leeren Trajektorien beseitigt. Da das trainierte Modell Strukturen erkennt, ist es nun möglich einen Fokus auf die Art der Struktur zu setzen und nicht durch die Anzahl an Punkten abgelenkt zu werden. Ein immernoch auftretendes Problem ist, dass Datenpunkte abgebildet werden, welche nicht zur selben Route des Schiffes gehören. Dadurch wird das Ergebnis verfälscht, da ein Schiff nach fahren einer Route den Hafen wechseln könnte und somit die Distanz auf dem Bild so sehr verändert, dass die eigentliche Route nur noch sehr klein bis gar nicht mehr erkennbar ist.

- **Filtern nach tripCount:** Hier wird die Trajektorie gleich einem TripCount gesetzt, wobei dadurch das Problem bis auf einige Fehler in den Trips beseitigt wird. Um nun eine gleiche Anzahl an Datenpunkten zu gewährleisten, kann ein Trip auf die ersten n Nachrichten verkürzt werden. Da trotz Cleaning noch große Sprünge in den Daten vorhanden sind, kommt es in den Bildern zum einem ähnlichen Problem wie beim Hafenwechsel eines Schiffes; Die eigentliche Route wird nicht mehr dargestellt, da der fehlerhafte Datenpunkt die Skalierung des Bildes dominiert. Ein solcher Fehler zeichnet sich bildlich durch eine gerade Linie durch einen Großteil des Bilds aus. Um solche Trajektorien trotzdem zu nutzen, ist eine Verfeinerung der Auswahl nötig.

- **Filtern nach tripCount und Skalierung[1]:** Die Trajektorie wird final gleich dem TripCount gesetzt, welcher innerhalb der ersten n Daten die geringste Skalierung besitzt. Dieser Ansatz umgeht das Problem verschiedener Trips und stellt weine gleiche Anzahl an Daten klar. Dadurch, dass jeweils die kleinste Skalierung ausgewählt wird, wird die Chance von fehlerhaften Daten minimiert. Diese Art des Filterings bietet auch in der Praxis sehr zuverlässige Ergebnisse, und liefert somit für fast alle der 36 Schiffe brauchbare Trajektorien.

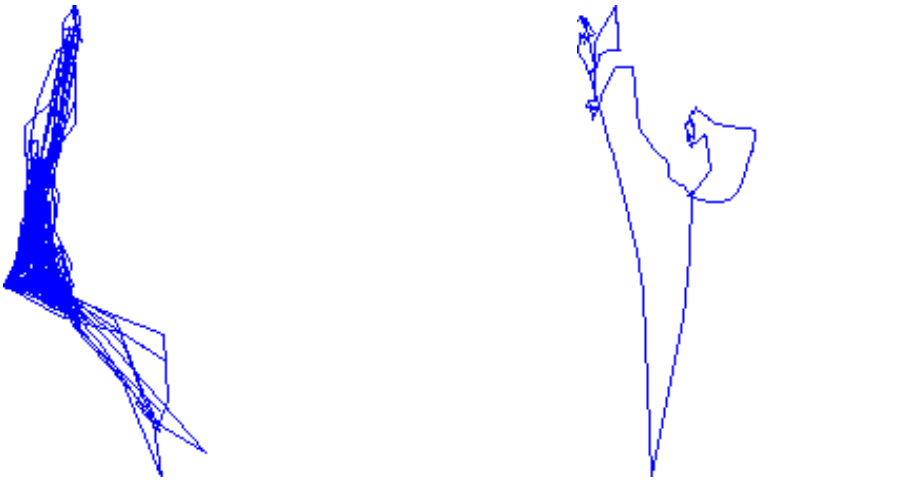
[1] Als Skalierung wird die größere der beiden geographischen Entfernungen beschrieben, welche eine Trajektorie in Längen- oder Breitengrad zurücklegt.

Durch händisches Selektieren von Ausführungen der obigen Filterings, stellte sich $n = 1.000$ als guter Richtwert der Anzahl der Trajektorien raus. Ausreißer werden gering gehalten, und die Trajektorien sind groß genug, um Routen zu erkennen.

Im Folgenden werden die verschiedenen Schritte in der Entwicklung der Bilder dargelegt. Dazu werden zwei Trajektorien als Beispiele dienen, an denen die Änderungen demonstriert und erläutert werden. Die Bilder und Daten sind Ergebnisse der Standarteinstellungen des Programms.

4.1.2 Grundkonstrukt

Um die Route eines Schiffes abzubilden, werden zunächst die Koordinaten in Pixel umgerechnet, und dann blaue Striche zwischen die Punkte gezeichnet. Der Informationsgehalt dieses Grundkonstruktes beschränkt sich also nur auf die Koordinaten. Alle Trajektorien werden auf quadratische Bilder abgebildet. Das spätere Image Embedding ermöglicht zwar einen Vergleich von Bildern mit verschiedenen Seitenrelationen, allerdings fokussiert sich das Modell sehr auf diesen Unterschied. Bei der quadratischen Implementation kann es sich somit auf die Struktur der Trajektorie fokussieren und der Vergleich des Seitenverhältnisses ist trotzdem noch möglich. Die Bilder haben alle eine Größe von 224 x 224 Pixeln, da das benutzte deep learning Modell keine höheren Bilder akzeptiert.



(a) Trajektorie 0 (b) Trajektorie 17

Abbildung 4.2: Liniendiagramm zweier Trajektorien

Auf beiden Bildern ist deutlich eine Route zu erkennen, wobei 4.2(b) den Anschein macht, weniger Datenpunkte zu haben. Tatsächlich besitzt 4.2(a) 1.000 und (b) 874 Datenpunkte, der visuelle Unterschied folgt aus einem großen Unterschied in der Skalierung. In (a) lässt sich links in der Mitte und bei (b) unten in der Mitte ein Hafen identifizieren. Schon hier lassen sich Unterschiede in der Route feststellen: (a) fährt erst zu einem Punkt, nach oben,

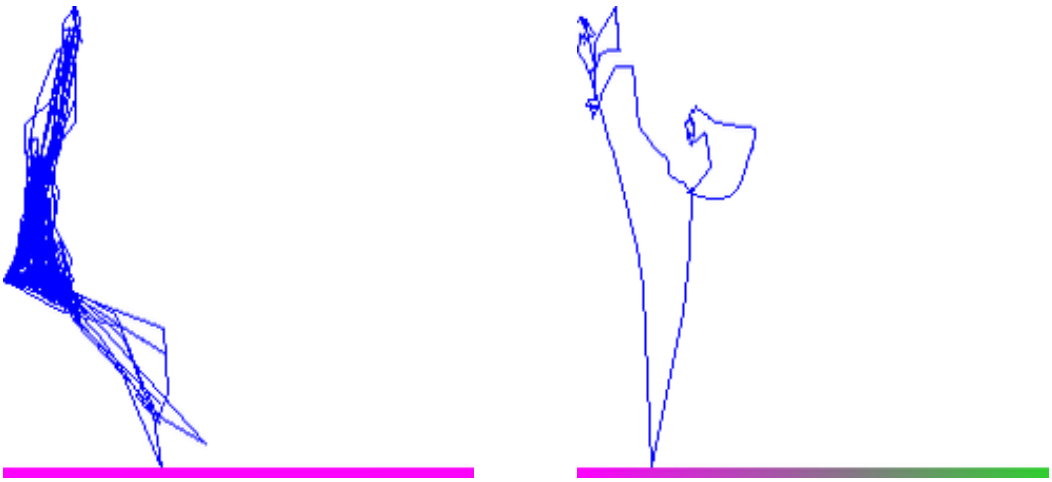
danach zurück zum Hafen, und danach zum zweiten Punkt, nach unten. (b) scheint erst alle Punkte abzufahren und anschließend zurück zum Hafen zu fahren.

4.1.3 Skalierung

Die Implementation der Skalierung ermöglicht dem Image Embedding Modell Zugriff auf Informationen über die geographische Distanz, welche die Schiffe zurücklegen.

Abbildung 4.3: Farbverlauf der Skalierung

Die Darstellung dieser Distanz erfolgt durch einen farblichen Verlauf am unteren Ende eines Bildes, ein Gradient. Ist die Distanz sehr gering, ist dieser fast ausschließlich Lila. Mit zunehmender Distanz färbt sich der Gradient nach rechts hin grün 4.3. Bevor die Skalierung einer Trajektorie ermittelt wird, muss zuerst durch den ganzen Datensatz iteriert werden, um die größte Skalierung zu finden. Das ist nötig, um die Farbwerte an die Distanz anzupassen, und das Feature somit adaptiv an das Filtering und den Datensatz zu machen. Die Skalierung wird anders als die Koordinaten im Grundkonstrukt noch in Seemeilen[2] umgerechnet, um falsche Daten durch Kartenprojektion zu vermeiden.



(a) Trajektorie 0 mit Skalierung

(b) Trajektorie 17 mit Skalierung

Abbildung 4.4: Liniendiagramm zweier Trajektorien mit farblich kodierter Skalierung

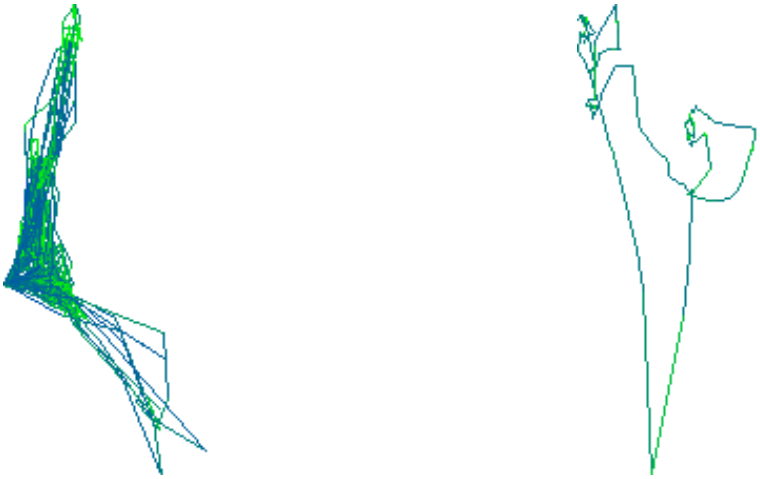
Die größte Skalierung über das ganze Datenset beträgt 1364sm. 4.4(a) legte eine Distanz von 20.23sm und 4.4(b) eine Distanz von 1108.05sm hin, womit (b) 81.25% des Farbspektrums eines vollen Gradienten abdeckt. Die Erklärung aus 4.1.3, dass die Anzahl der Punkte durch die Skalierung getäuscht wird, wird mit diesem Feature bestätigt.

[2]Eine Seemeile wird mit sm abgekürzt und steht umgerechnet für 1.852km.

4.1.4 Geschwindigkeit

Abbildung 4.5: Farbverlauf der Geschwindigkeit

Die Geschwindigkeit eines Schiffes wird durch die Farbe der Linie zwischen zwei Datenpunkten dargestellt. Dabei wird wieder im Vorhinein der größte Wert aller Geschwindigkeiten ermittelt, damit sich das Programm auf diese anpassen kann. Mit steigender Geschwindigkeit steigt der Blauton, und der Grünton nimmt linear ab 4.5. Beträgt eine Geschwindigkeit an zwei Punkten die größte Geschwindigkeit, ist die Farbe der Linie zwischen diesen Punkten also (0,0,255). Beträgt die Geschwindigkeit genau der Hälfte, so ist die Farbe (0,127,127).



(a) Trajektorie 0 mit Geschwindigkeit

(b) Trajektorie 17 mit Geschwindigkeit

Abbildung 4.6: Liniendiagramm zweier Trajektorien mit farblich kodierter Geschwindigkeit

Die maximale Geschwindigkeit des Datensatzes beträgt 11.7kn[3], Wobei die von 4.6(a) 9.6kn und die von 4.6(b) 7.2kn beträgt. Beide zeichnen sich durch eine höhere Geschwindigkeit am Hafen, und eine geringere an den vermeitlichen Ausrüstungsorten aus. Durch die Geschwindigkeit lässt sich nun besser der Ort einschätzen an dem das Schiff zum Fischen hält. (b) scheint nach dieser Information viele Stops zu haben, da die Geschwindigkeit in ähnlicher Frequenz variiert. Das passt mit der hohen Distanz nach 4.4 gut zusammen, da auf hoher Distanz viele Ausrüstungen aufgestellt sein können.

[3] 1 Knoten entspricht 1.852km/h

4.1.5 Wassertiefe

Als letztes Feature wird die Wassertiefe in das Bild kodiert. Hierbei gibt es zwei verschiedene Ausführungen: Kodierung durch farbige Kreise hinter den Datenpunkten und eine Karte im Hintergrund der Route.

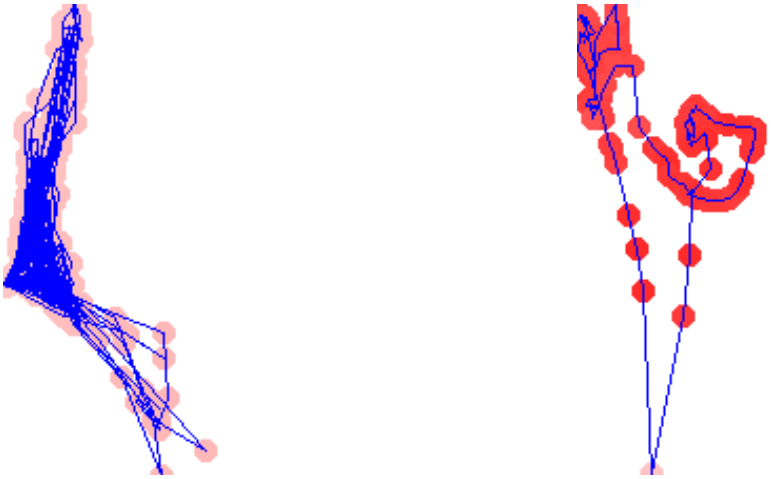
Wassertiefe mit Punkten

Diese Darstellung stützt sich auf die Spalte *waterDepth* aus dem cleaned Dataset. Durch diese Spalte ist es möglich die Wassertiefe der vorhandenen Koordinaten zu erhalten, um diese dann hinter die bereits im Grundkonstrukt gezeichneten Linien zu kodieren.

Abbildung 4.7: Farbverlauf der Wassertiefe

Auch dieses Feature wird wieder durch einen Verlauf der Farben visualisiert, wobei dafür die höchste Wassertiefe aller Trajektorien ermittelt werden muss. Bei steigender Wassertiefe eines Punktes und geringer werdendem Abstand zur höchsten Wassertiefe sinken die Grün- und Blauwerte der Punkte 4.7. Eine sehr große Wassertiefe bedeutet ein sehr roter Punkt mit dem RGB-Wert (0,0,255), wobei eine Wassertiefe von 0 für einen blassen roten Punkt mit dem RGB-Wert (255,204,204) steht.

Der Verlauf zwischen diesen Werten verläuft nicht linear, denn die jeweilige Wassertiefe w wird mit der Funktion $\sqrt[3]{w^5}$ gewichtet. Mit dieser Gewichtung kann besser ein Unterschied zwischen flachen Wassertiefen festgestellt werden, da aufgrund der extremen Unterschiede bei einer linearen Gewichtung fast nur zwischen sehr flach und sehr tief unterschieden wird.



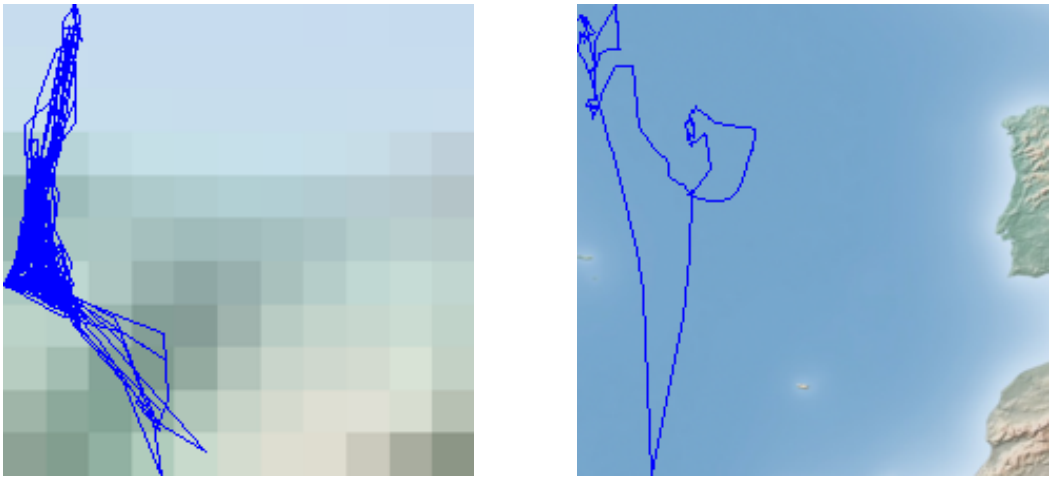
(a) Trajektorie 0 (b) Trajektorie 17

Abbildung 4.8: Liniendiagramm zweier Trajektorien mit farblichen Punkten als Wassertiefe

Die größte Wassertiefe beträgt 5643m, wobei die größte von 4.8(a) 147m und von 4.8(b) 4993m beträgt. Durch die Punkte ist die Anzahl an Nachrichten pro Trajektorie noch besser sichtbar, wodurch deutlich wird, dass (b) auf dem vermutlichen Hinweg wenige Nachrichten gesendet hat, da dort wie in 4.6 zu sehen weniger Zeit vergangen ist. Die Punkte von (a) sind vorhanden, durch die maximale Tiefe von 147m erscheinen diese allerdings sehr blass. Desweiteren wird die Vermutung der Orte der Häfen bestätigt, da an beiden Orten blasse Punkte zu sehen sind, was nur gilt wenn $w \leq 0$ gilt, die Punkte also am Festland liegen müssen.

Wassertiefe mit Karte

Die Implementierung der Wassertiefe über eine Karte im Hintergrund basiert auf die Python Bibliothek Matplotlib, welche unter anderem eine Reihe an *Basemaps* bereitstellt. Diese können anhand der in den Ecken liegenden Koordinaten extrahiert und als Bild dargestellt werden. Die in dieser Implementation gewählte Basemap wird durch die Funktion *shadedrelief()* dargestellt, und kann die Wassertiefe in verschiedenen Blautönen darstellen. Zusätzlich wird auch das Festland in verschiedenen Farben dargestellt. Diese Art der Basemap ist ein Foto, was einen riesigen Detailverlust bei kurzer Skalierung mit sich bringt.



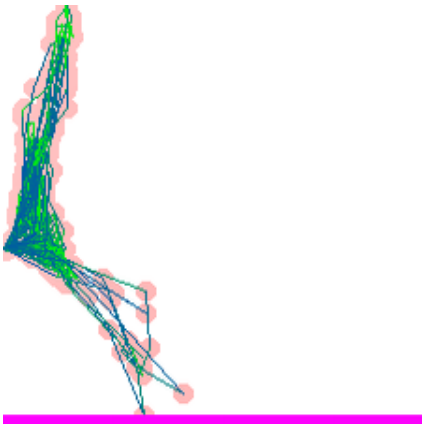
(a) Trajektorie 0 mit Wassertiefe als Karte

(b) Trajektorie 17 mit Wassertiefe als Karte

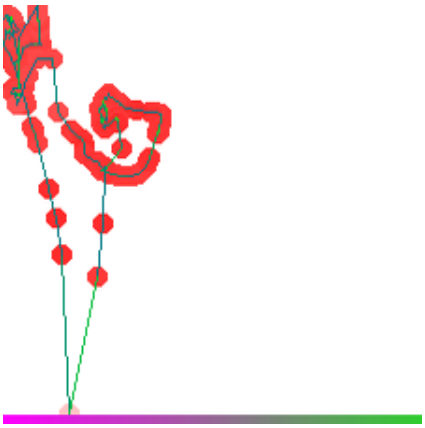
Abbildung 4.9: Liniendiagramm zweier Trajektorien mit Darstellung der Wassertiefe als Karte

Die Wassertiefe ist bei 4.9(a) erkennbar kleiner als bei 4.9(b). Da es sich um ein Foto handelt, befinden sich auf der unteren Hälfte von (a) bereits Farbtöne welche auf Festland schließen lassen. Der unterschiedliche Detailgrad gibt dem Image Embedder weitere Auskünfte über die Skalierung, allerdings könnte das Modell dazu neigen einen großen Fokus auf das Festland zu legen, welches wenig Informationen über die Fixed Gear Klasse enthält.

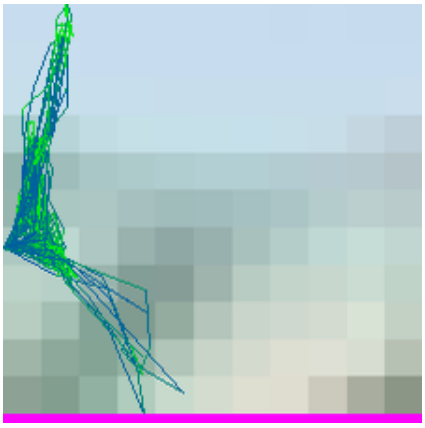
4.1.6 Alle Features



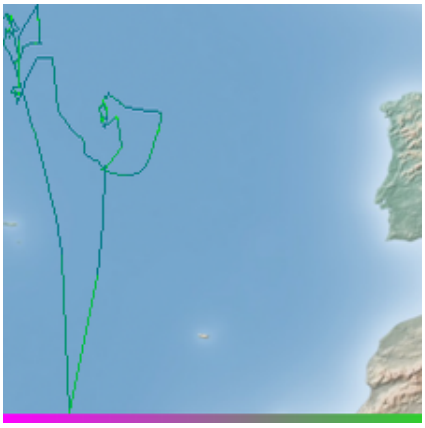
(a) Trajektorie 0 mit allen Features und Punkten als Wassertiefe



(b) Trajektorie 17 mit allen Features und Punkten als Wassertiefe



(c) Trajektorie 0 mit allen Features und Karte als Wassertiefe



(d) Trajektorie 17 mit allen Features und Karte als Wassertiefe

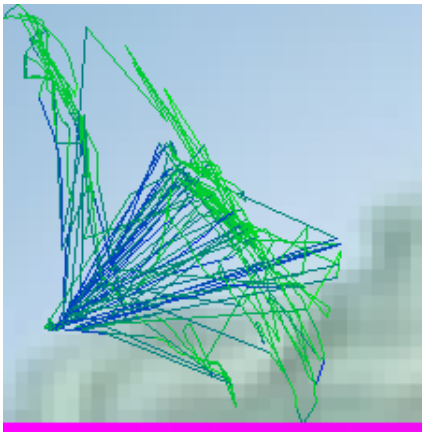
Abbildung 4.10: Liniendiagramme von Trajektorien mit allen implementierten Features

Mit bloßem Auge lässt sich in Abb. 4.10 ein klarer Unterschied zwischen beiden Trajektorien erkennen. So fährt das Schiff der Trajektorie 25 weit auf das Meer hinaus und fischt in tiefen Gewässern, was ein Indikator für die Langleinenfischerei ist. Die im Vergleich viel flacher aufgebaute Ausrüstung deutet dabei auf Potting/Trapping hin. Die Skalierung beider Trajektorien bestätigt diese Vermutung.

4.1.7 Trawlers

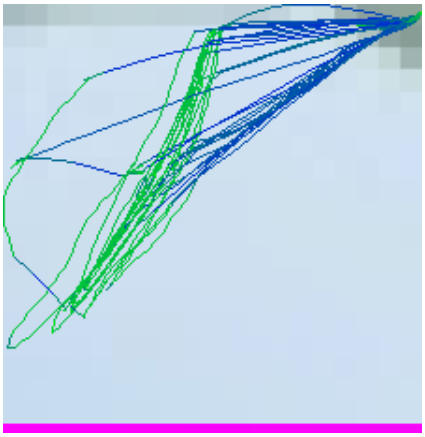
Im Clustering werden zusätzlich vier Trawler Trajektorien aufgenommen. Ziel ist es hierbei, herauszufinden, ob das Modell einen Unterschied zu einem in der Theorie sehr großen Kontrast zu der Fixed Gear Klasse erkennt. Da der Datensatz der Trawlernachrichten keine

gesäuberten Daten mit zusätzlichen Informationen enthält, ist das Zeichnen der Wassertiefe über Punkte nicht möglich. Außerdem enthält dieser Datensatz viele Ausreißer in den Daten, was dazu führt, dass händisch vier Trajektorien ohne Ausreißer ausgewählt wurden. Die Werte der Skalierung und der Geschwindigkeit gingen hierbei in der Berechnung der größten Werte mit ein und berücksichtigen diese auch.



(a) Trawler 0

(b) Trawler 1



(c) Trawler 2

(d) Trawler 3

Abbildung 4.11: Ausgewählte Trawler Trajektorien mit allen Features und Karte als Wassertiefe

Die Skalierung der dargestellten Trawlertrajektorien in Abb. 4.11 beträgt zwischen 36sm und 135.255sm, wobei letzteres 9.92% der höchsten Skalierung ausmacht. Das ist ein großer Unterschied zu manchen Fixed Gear Trajektorien, und es bestätigt die Theorie in 2.3.2, dass Trawler in Küsten Nähe fischen, da die Wassertiefe dort gering ausfällt. Desweiteren lässt sich in allen Vier Bildern ein Muster erkennen; Die Schiffe fahren hinaus und fischen in geringerer Geschwindigkeit, wobei sie längere Distanzen fahren, bevor sie wieder zurück in den Hafen fahren.

4.1.8 Ausführung

Die Klasse Draw.py kann mit *python.exe Draw.py* gestartet werden. Hierbei wird das Programm mit den Standartparametern geladen. Die Klasse erzeugt im Programm befindenden Ordner einen weiteren Ordner *drawedImages*, indem die erstellten Bilder nach dem gespeichert werden.

```
greatestScale: 22.71429
greatestspeed: 11.7
greatestwaterDepth: -5643.0

mmsi: 30581813007233.0
length of trajectory: 1000
imageScale: 0.33697
max Speed of trajectory: 9.6
image 0 drawed in 0.357739 seconds
```

Abbildung 4.12: Ein Ausschnitt der drawing.log Datei

Zudem erzeugt das Programm die Datei *drawing.log*, welche Informationen über den ganzen Datensatz sowie den gemalten Bildern speichert, um diese auch noch im Nachhinein nachvollziehen zu können. Desweiteren stellt die Klasse eine Reihe an Parametern zu Verfügung. So kann mit *'–DATASET raw'* das Programm mit dem rohen Datensatz geladen werden. Hinzufügen von *'–FILTERING time'* filtert die Trajektorien wie im Kapitel Filtering erwähnt. Um rohes und gesäubertes Datenset, sowie die verschiedenen Filterarten vergleichen zu können, wird je nach Ausführung ein eigener Ordner erstellt. Eine Dokumentation zu allen Parametern kann mit *python.exe Draw.py -help* erzeugt werden.

4.2 Fine Tuning

Um das bereits trainierte *ResNet50* Modell auf die genaue Aufgabe dieser Arbeit zu fokussieren, wird *transfer Learning* angewendet. Diese Aufgabe wird in der Klasse *fineTune.py* gelöst und benutzt die Python Bibliothek *TensorFlow*, um Operationen auf dem Modell ausführen zu können.

Die Trainingsdaten setzen sich aus dem Fixed Gear Datensatz, sowie Datensätze anderer Fischfangtechniken zusammen, welche ebenso von Global Fishing Watch stammen und dem selben Datenbankschema folgen. Auf diesen Daten wird Draw.py mit verschiedenen Einstellungen an Features aufgerufen, um eine höhere Anzahl an Trainingsdaten bereitzustellen und das Modell möglichst genau auf die Disziplin zu trainieren. Die erhaltenen Bilder wurden händisch auf starke Fehler überprüft und gegebenenfalls ausgefiltert. Daraus resultierten insgesamt 726 Bilder, von denen 351 zu *Drifting Longlines*, 148 zu *Trawler*, 130 zu *Fixed Gear* und 97 zu *Purse Seiners* gehören. Die Trainingsdaten sind in dem Ordner *./data/training_images* aufzufinden.

Das Programm lädt ResNet50 ohne die bestehende Ausgangsschicht ein, und friert vor dem Trainin alle vorhandenen Schichten ein, um das Training dieser zu vermeiden und somit

eine schnellere Laufzeit zu ermöglichen. Daraufhin wird eine eigene Schicht erstellt: *Global Average Pooling(GAP)*. GAP berechnet den Durchschnitt der Features aus allen Feature Maps des vorherigen Layers, wodurch jeder einzelnen Feature Map ein Wert zugeordnet wird. Resultat aller Werte ist der Feature Vektor. Anschließend folgt das Training des Modells in 18 Epochen.

4.3 Metrik

Im Anschluss der Erstellung aller Bilder und dem Transfer Learning, wird nun die Distanzmatrix erstellt. Diese ist eine zweidimensionale Liste, welche jedem Schiff die Distanz zu allen anderen, einschließlich sich selbst, zuweist. Metric.py lädt alle Bilder ein, welche sich im eingestellten Dateipfad, standardweise *./data/drawedImages/cleaned/adaptive/*, befinden. Anhand der Einbettung der Bilder von ResNet50 wird zunächst die Ähnlichkeit *ss* zwischen den Bildern, mittels Cosine Similarity berechnet. Die Distanz zwischen den Bildern, welche für die Distanzmatrix benötigt wird, wird durch $1 - s$ berechnet.

Die Klasse Metric.py ist kein eigenständig auszuführendes Programm, da es die Matrix zur weiteren Verarbeitung zur Verfügung stellt. Sie wird von Cluster.py importiert und mit dem Python Objektparameter *.metric* kann auf das Feld der Metrik selbst zugegriffen werden.

4.4 Clustering

4.4.1 Umsetzung

Die Implementierung des Clusterings von k-means stützt sich auf die Bibliothek *sklearn*. Die Anzahl der Klassen wird $k = 4$ gesetzt, welche sich aus 3 verschiedenen Fixed-Gear Fangtechniken und einer Trawler Klasse ergibt. Für eine einheitliche Berechnung wird der $random\_state = 27$ gesetzt, da der Algorithmus zufallbasiert ist und somit jede Berechnung wiederholt werden kann.

Zur zweidimensionalen Darstellung des Clusterings in einem Streudiagramm, wie in Abb. 2.2, wird zunächst PCA als Werkzeug der Dimensionsreduzierung herangezogen. Die dadurch erhaltene Liste dient dem Streudiagramm als Input. Zur Markierung der in k-means zugeordneten Klasse, wird jedem Datenpunkt eine Farbe gegeben und um auch die Trawler sichtlich zu unterscheiden, werden diese als Dreiecke markiert.

4.4.2 Ausführung

Das Programm ist in Cluster.py geschrieben und kann mit *python.exe Cluster.py* ausgeführt werden. Es importiert die Klasse Metric.py. Mit –datapath <Datapath> kann der Pfad zu den zu analysierenden Bildern festgelegt werden und auch der random_state wird mit –seed <seed> festgelegt. Eine umfassende Dokumentation über alle Parameter wird mit *python.exe Cluster.py -help* aufgerufen.

5 Evaluation

Dieses Kapitel dient der Auswertung der einzelnen Features der Implementation. Die Auswertung wird durch Streudiagramme und den jeweiligen Silhouettenkoeffizienten unterstützt. Um Unterschiede in den Entscheidungen des Modells zu verdeutlichen werden sich Einzelfälle genauer angeschaut. Die in 4.1.7 aufgezeichneten Trawler dienen als Kontrast zu den Fixed Gear Bildern und sollten somit immer in einer eigenen Klasse landen.

5.1 Auswertung

5.1.1 Grundkonstrukt

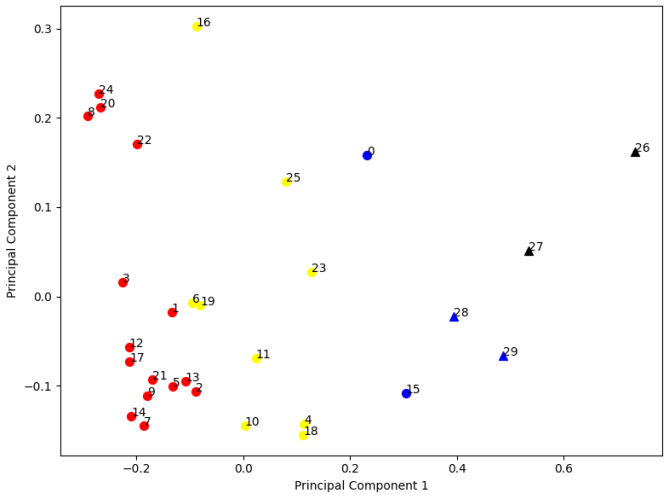
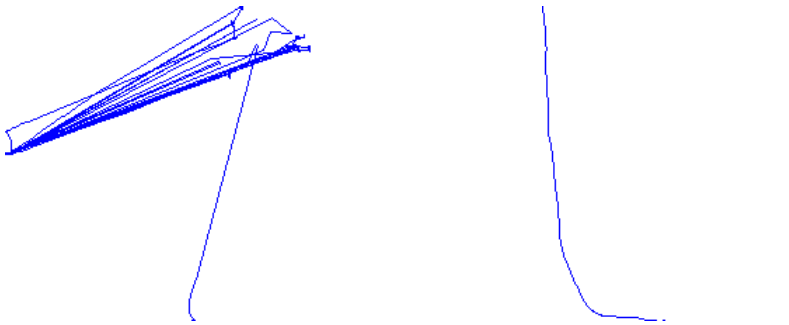


Abbildung 5.1: Streudiagramm des Grundkonstrukts der Trajektorien

$S_C = 0.2201$

Das Clustering der Grundkonstrukte 5.1, also die Darstellung der Route als Linien zwischen den Punkten, weist eine ungleichmäßige Verteilung auf. So sind 15 Bilder der roten und nur 2 Bilder der schwarzen Klasse zugeteilt worden. Diese Ungleichmäßigkeit ist allerdings möglich, da es keine Auskunft über die tatsächliche Verteilung im Datensatz gibt. Die 4 Trawler wurden nicht der selben Klasse zugeteilt, allerdings stehen sie sich im Diagramm sehr nah. Das Modell scheint also einen Unterschied zwischen Trawler und Fixed Gear erkennen zu können, was womöglich an den gut zu erkennenden Linien liegt, welche in den meisten Fixed Gear Bildern aufgrund der Skalierung nicht auftritt.



(a) Trajektorie 23. (b) Trajektorie 24.

Abbildung 5.2: Liniendiagramme zweier Trajektorien

Die vier roten Datenpunkte in der oberen linken Ecke weisen alle eine ähnliche Struktur auf, die sich meist durch eine wie in 5.2 (a) durch das Bild führende Linie auszeichnet. Alle anderen roten Punkte bestehen aus ähnlichen Linien, mit allerdings einigen Abzweigungen. Die gelbe Klasse hat bereits mehr erkennbare Linien 5.2 (b) und die blaue besteht aus zwei Trawlern und zwei sehr ähnlich aussehenden Fixed Gear Bildern. Mit steigendem PC1 steigt im Grundmodell die Deckkraft der Bilder, wobei mit steigendem PC2 in den meisten Fällen die Verzweigungsanzahl sinkt.

5.1.2 Skalierung

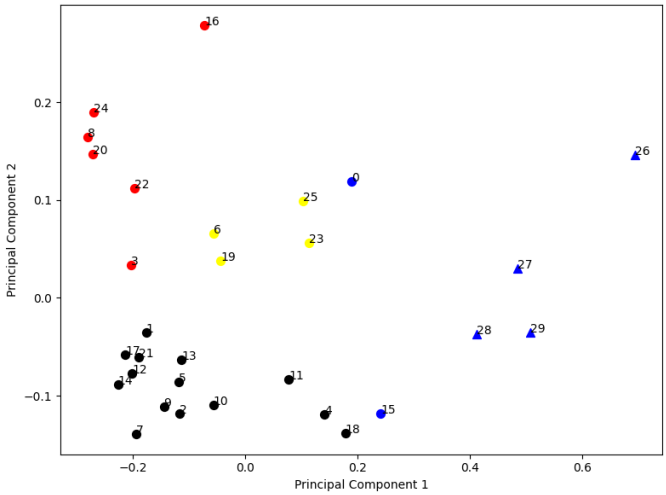


Abbildung 5.3: Streudiagramm aller Trajektorien mit farblich kodierter Skalierung

$S_C = 0.2675$

Im Vergleich zum Streudiagramm des Grundkonstrukts 5.1 treten in Abb. 5.3 bezüglich der Koordinaten keine großen Veränderungen auf. Die Punkte wandern tendenziell in

Richtung ihrer Klassenzentren, wodurch die Klassen etwas isolierter werden, was auch in der leichten Steigerung von S_C deutlich wird. Die Aufteilung in den Klassen verzeichnet eine größere Änderung; Die noch im Grundkonstrukt eher kleinen Klassen Blau und Schwarz werden hier zusammengefügt, wodurch nun alle Trawler beisammen sind. Die Punkte 3, 4, 5, 8, 12, 17 und 22 sind die einzigen Bilder, welche sich mit einer Skalierung von über 50% der größten auszeichnen, welche durch das Hinzufügen des Features allerdings nur ein wenig zueinander rücken. Alle diese Punkte sind in den Klassen rot oder schwarz, wodurch die Vermutung aus Abschnitt 5.1.1 bestätigt wird.

5.1.3 Geschwindigkeit

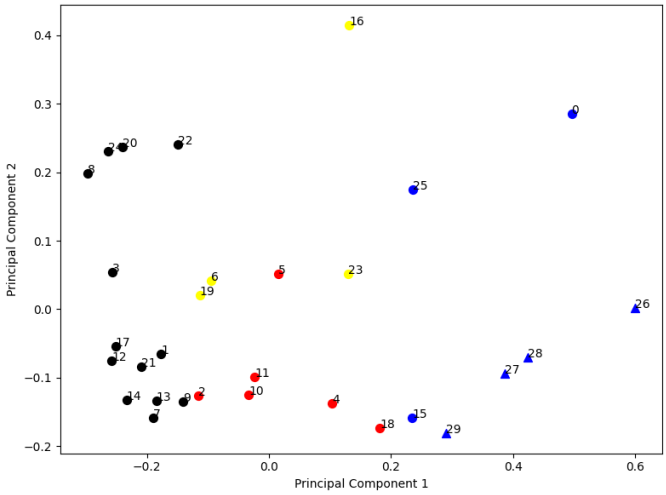


Abbildung 5.4: Streudiagramm aller Trajektorien mit farblich kodierter Geschwindigkeit

$S_C = 0.2015$

(a) Trajektorie 5.

(b) Trajektorie 25.

Abbildung 5.5: Liniendiagramm zweier Trajektorien mit farblich kodierter Geschwindigkeit

In Abb. 5.4 ist zu erkennen, dass Trajektorie 25 nun der blauen Klassen zugeordnet wird, welche wie die anderen Trawler und Bild 0 und 15 viel Platz einnehmen, und sich durch starken Konstrast in den Farben der Linien zwischen dem vermeitlichen Häfen und Ausrüstungspunkten auszeichnen. Dieser Kontrast wird in Abb. 5.5(b) deutlich. Diese Klasse spiegelt vermutlich die Menge der Trawler und Potting Schiffen wieder, da diese auch durch die Skalierung viel Platz einnehmen, im Gegensatz zu den Trawlern allerdings eine höhere Geschwindigkeit zwischen den Punkten der Ausrüstungsorte haben. Die in Abschnitt 5.1.1 noch rot gewesene Klasse ist nun schwarz und enthält bis auf 2 und 5 die selben Punkte. Punkt 5 ist der einzige weit gewanderte Punkt dieser Klasse, was womöglich an der zu der Geschwindigkeit ähnlichen Eigenschaft der blauen Klasse liegt, was in einem Vergleich der Farben in den Abbildungen 5.5(a) und 4.11 zu erkennen ist.

5.1.4 Wassertiefe

Wassertiefe als Punkte

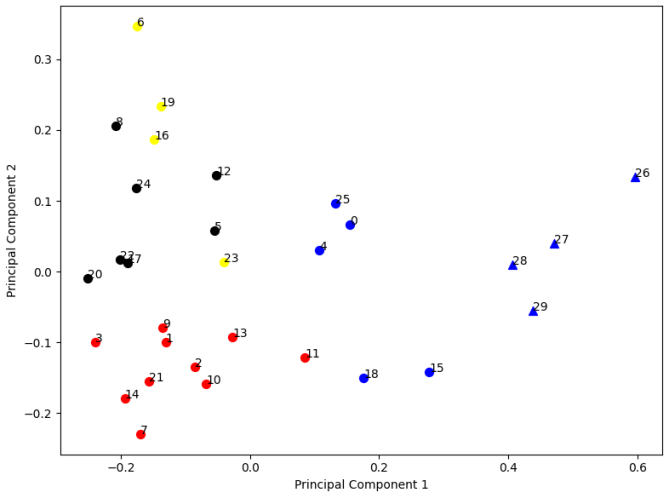


Abbildung 5.6: Streudiagramm aller Trajektorien mit Darstellung der Wassertiefe als Punkte

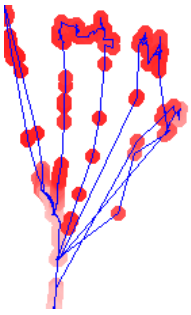


Abbildung 5.7: Liniendiagramm von Trajektorie 4 mit Darstellung der Wassertiefe als Punkte

$S_C = 0.2165$

In Abb. 5.6 wird eine positive Korrelation zwischen PC2 und der Wassertiefe deutlich. Ein gutes Beispiel dafür ist Bild 4 (Abb. 5.7); Dieses behält seinen PC1 Wert, und wandert 0.3 in positiver Ordinatenrichtung. Durch den selben Grund werden die in Abb. 5.1.1 noch sehr eng beieinander liegenden aus dünnen Linien bestehenden Bilder durch die Informationen des neuen Features auseinander gerückt. Das Modell ordnet die Informationen der Wassertiefe richtig ein und beschränkt sich nicht nur auf diese. Es werden also Muster und Tiefe der Trajektorien berücksichtigt.

Wassertiefe als Karte

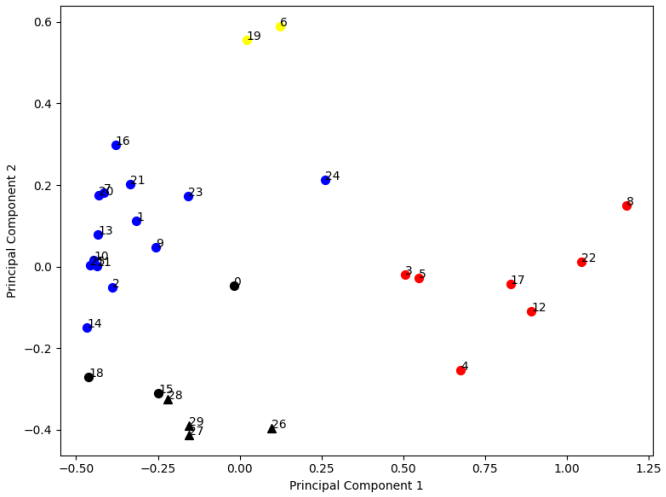


Abbildung 5.8: Streudiagramm aller Trajektorien mit Darstellung der Wassertiefe als Karte

$S_C = 0.2208$

Abb. 5.8 visualisiert, was der etwas höhere S_C beschreibt: Eine klarere Trennung der Klassen. Diese wird zudem noch durch eine größere Skalierung im Diagramm selbst verstärkt. Die rote Klasse, dessen Punkte zuvor der roten oder gelben Klasse angehörten, haben alle bis auf 3 und 5 gemeinsam, dass sie Trajektorien im Atlantik vor Portugal beinhalten. Desweiteren haben sie auch ein ähnliches Muster, das aus einem langen Weg zu den Ausrüstungspunkten hin, kurzen Wegen zwischen diesen, und einem langen Weg zurück besteht. Das Modell entscheidet also nicht nur nach der Skalierung, welche hier insbesondere durch die Auflösung ermittelt werden kann, sondern auch nach den Mustern der Trajektorien.

(a) Trajektorie 4

(b) Trajektorie 22

Abbildung 5.9: Liniendiagramme zweier Trajektorien mit Darstellung der Wassertiefe als Karte

Die Ähnlichkeit dieser Muster ist gut in Abb. 5.9(a) und Abb. 5.9(b) zu erkennen. So sieht es aus, als wäre (b) ein Teil von (a), was aus der Anzahl an Punkten in (a) resultiert. Die Trawler liegen erneut eng beieinander und bilden zusammen mit anderen Bildern mit niedrig auflösenden Karten im Hintergrund die schwarze Klasse. Der Verlgeich zu Abschnitt 5.1.4 fällt relativ starkt aus. So liegen zwar erneut alle Trawler beieinander, das Modell konzentriert sich allerdings weniger auf die vorher noch deutlicher zu erkennende Anzahl an AIS Nachrichten.

5.1.5 Alle Features

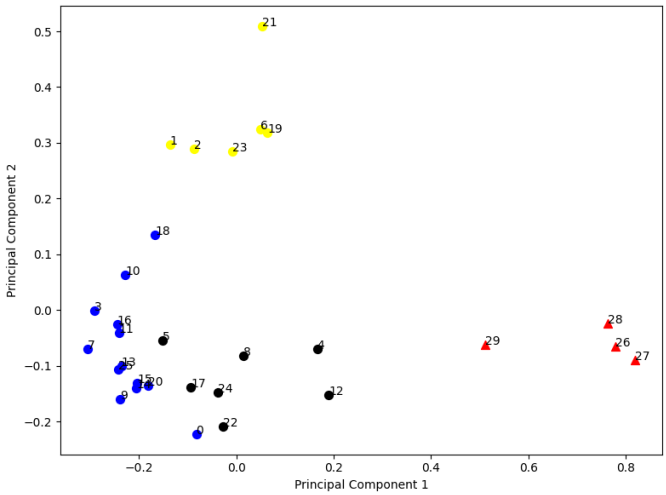


Abbildung 5.10: Streudiagramm aller Trajektorien mit allen Features und Darstellung der Wassertiefe als Punkte

$S_C = 0.2165$

Abb. 5.10 zeigt, dass durch die Codierung aller Features in die Bilder, das Modell die 4 Trawler in eine eigene Klasse ordnet, welche zudem noch die isolierteste aller Klassen ist. In gelb wird eine Klasse markiert, welche tendenziell durch eine mittlere Wassertiefe, mittlere Skalierung und hohe Geschwindigkeit ausgezeichnet wird. Blau und Schwarz fangen dabei Trajektorien auf, welche Eigenschaften beider vorherigen Klassen enthalten. Hier scheint das Modell nicht gut mit der Summe an Informationen umgehen zu können. Die vorherigen Ergebnisse drängen die Punkte tendenziell alle in unterschiedlich Richtungen, und die hier stattfindende Kombination der Informationen verursacht, dass das Modell keine klare Entscheidung treffen kann, wodurch das Modell in dieser Ausführung nur gering die theoretischen Grundlagen wiedergibt.

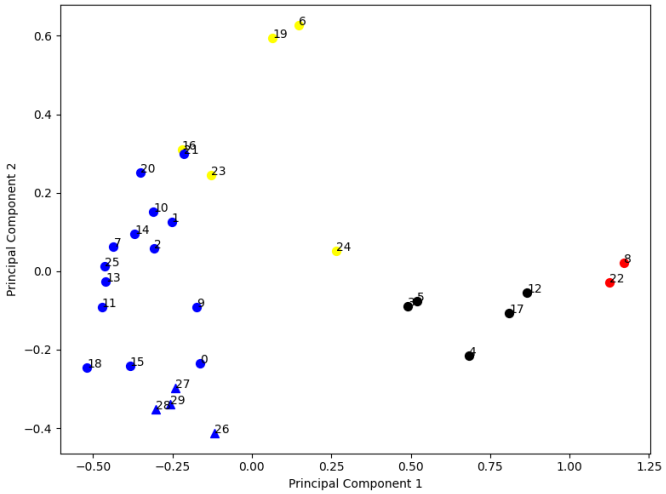


Abbildung 5.11: Streudiagramm aller Trajektorien mit allen Features und Darstellung der Wassertiefe als Karte

$S_C = 0.3032$

Der Unterschied zwischen Karte und Punkten im Hintergrund beeinflusst die Distanzberechnung des Modells stark. Die Trawler nähern sich wie in Abb. 5.9 den anderen Punkten an und sind somit Teil einer sehr großen Klasse. Die Größe der blauen Klasse in Abb. 5.11 impliziert den bisher größten Silhouettenkoeffizienten, da für den Großteil aller Punkte der Abstand zum eigenen Klassenmittelpunkt geringer ist, als zu anderen. Das Problem aus 5.10 tritt hier etwas stärker auf. Allerdings ist der erwartete Effekt der Konzentration auf den Hintergrund und die Küstenlininen geringer als erwartet. Dies kann aber auch mit dem kleinen Datensatz zusammenhängen, denn dieser enthält Trajektorien der selben Klasse mit ähnlichen Koordinaten und somit ähnlichem Hintergrund enthält.

5.2 Fazit und Ausblick

Die Anwendung des trainierten ResNet50 Modells zur Berechnung der Distanzmatrix liefert dem Clusteringsalgorithmus k-means einen großen Informationsgehalt der Daten der Trajektorien. Werden dem Modell einzelne Features präsentiert, so bezieht es diese mit in die Distanzberechnung ein und liefert Ergebnisse, welche bis auf Ausnahmen die Theoretischen Grundlagen wiederspiegeln. Eine Kombination aller Features liefert bei beiden Implementationsarten der Wassertiefe ein von der Theorie abweichendes Ergebnis. Das Modell kann in allen Arten der Featurekombination gut eine Ähnlichkeit zwischen Trawlern unter sich erkennen, sowie einen gewissen Unterschied zu den Fixed Gear Trajektorien, welche insbesondere bei der Skalierung, sowie der Darstellung der Wassertiefe als Punkte auftritt.
Um die Aussagekraft des Features der Wassertiefe als Karte zu verstärken, wäre eine andere Darstellungsart hilfreich. So könnte besser zwischen den Wassertiefen differenziert werden, und das Modell würde weniger Fokus auf die Küstenlinien legen.
Eine große Schwierigkeit der Aufgabe war der Datensatz als Grundlage. So sind die berechneten Clusterings Resultate aus den Daten von nur 25 Fixed Gear Schiffen. Um die Aussagekraft des Modells zu erhöhen, ist die Anzahl an Schiffen zu erhöhen. Desweiteren ist der benutzte Datensatz nicht komplett von fehlerhaften Daten befreit, was sich bei dieser Art der Implementation als große Schwierigkeit herausstellte, da ein Bild durch einen großen Sprung die Aussagekraft verliert. Die Wahl einer geeigneten Trajektorie wurde durch den TripCount unterstützt, allerdings besteht auch dieser aus fehlerhaften Daten, wodurch zwei gleiche Fischfangtechniken im selben Gewässer auch unterschiedlich dargestellt werden, was in Abschnitt 5.9 beschrieben wurde. Durch das Anonymisieren der Schiffsdaten fehlt es dem Datensatz an statischen Daten wie der Schiffsgröße. Diese könnte als weiteres Feature dienen.
Auch das ResNet50 Modell würde von einem größeren Datensatz profitieren, da dadurch auch die Anzahl der Trainingsdaten des Transfer Learnings erhöht wird, was ein besseres Verständnis des Modells impliziert.

Eine ähnliche Idee der Distanzmatrixberechnung folgt aus der Darstellung einer Trajektorie als Animation. So würde sich das Schiff im Verlauf des Videos auf die aktuelle Position bewegen, wodurch die Geschwindigkeit durch die Dimension der Zeit dargestellt wird. Ziel dieser Implementierung wäre das Vermeiden des Auftretens von der unklaren Entscheidung des Modells, welche in Abb. 5.10 und Abb. 5.11 deutlich wird, und somit alle Informationen darstellen zu können.

Literaturverzeichnis

[Ali23] ALIZADEH, Esmaeil: *An Illustrative Explanation to Dynamic Time Warping*. https://towardsdatascience.com/an-illustrative-introduction-to-dynamic-time-warping-36aa98513b98. Version: Jul 2023

[BFDK23] BAYER, Mirjam ; FRY, Tabea ; DETHLEFSEN, Sören ; KAZEMPOUR, Daniyal: Pipeline for Open AIS Data with Filtering Based on Vessel Class. (2023)

[BTTT18] BIAN, Jiang ; TIAN, Dayong ; TANG, Yuanyan ; TAO, Dacheng: A survey on trajectory clustering analysis. (2018), Feb, Nr. arXiv:1802.06971. http://arxiv.org/abs/1802.06971. – arXiv:1802.06971 [cs]

[don14] DONALDSON lane: *PPT - Longest Common Subsequence Problem and Its Approximation Algorithms PowerPoint Presentation - ID:6196810*. https://www.slideserve.com/lane-donaldson/longest-common-subsequence-problem-and-its-approximation-algorithms. Version: Nov 2014

[GAKJ22] In: GUPTA, Yash ; ANKIT ; KULKARNI, Sanchit ; JAIN, Pooja: *Handwritten Signature Verification Using Transfer Learning and Data Augmentation*. 2022. – ISBN 9789811671357, S. 233–245

[GDD+19] GOLDER, Anupam ; DAS, Debayan ; DANIAL, Josef ; GHOSH, Santosh ; SEN, Shreyas ; RAYCHOWDHURY, Arijit: *Practical Approaches Towards Deep-Learning Based Cross-Device Power Side Channel Attack*. 2019 https://www.researchgate.net/publication/334288574_Practical_Approaches_Towards_Deep-Learning_Based_Cross-Device_Power_Side_Channel_Attack

[kme23] *ML | K-means++ Algorithm*. https://www.geeksforgeeks.org/ml-k-means-algorithm/. Version: Apr 2023

[Mar20] MARITIMEMANUAL: *Automatic Identification System*. https://www.maritimemanual.com/automatic-identification-system-ais/. Version: Mar 2020

[Mar21] MARINETRAFFIC: *What is the Automatic Identification System (AIS)?* https://help.marinetraffic.com/hc/en-us/articles/204581828-What-is-the-Automatic-Identification-System-AIS-. Version: Aug 2021

[Med23] MEDIAPIPE: *Image embedding task guide*. https://developers.google.com/mediapipe/solutions/vision/image_embedder. Version: Sep 2023

[Mok21] MOKHTARANI, Shabnam: *Embeddings in Machine Learning*. https://www.featureform.com/post/the-definitive-guide-to-embeddings. Version: Aug 2021

[MT09] MORRIS, Brendan ; TRIVEDI, Mohan: Learning Trajectory Patterns by Clustering: Experimental Studies and Comparative Evaluation. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition* 0 (2009), Jun, S. 312–319. http://dx.doi.org/10.1109/CVPRW.2009.5206559. – DOI 10.1109/CVPRW.2009.5206559. – ISSN 978–1–4244–3992–8

[Mwi22] MWITI, Derrick: *Transfer Learning Guide: A Practical Tutorial With Examples for Images and Text in Keras*. https://neptune.ai/blog/transfer-learning-guide-examples-for-images-and-text-in-keras. Version: Juli 2022

[Pra18] PRABHAKARAN, Selva: *Cosine Similarity - Understanding the math and how it works? (with python)*. https://www.machinelearningplus.com/nlp/cosine-similarity/. Version: Oct 2018

[R22] R, Adriano: *K-means: Fokus auf diesen Clustering Machine Learning Algorithmus*. https://datascientest.com/de/was-ist-k-means. Version: Dec 2022

[Tre16] TREVINO, Andrea: *Introduction to K-means Clustering*. https://blogs.oracle.com/ai-and-datascience/post/introduction-to-k-means-clustering. Version: Dec 2016

[Vic20] VICK, Alan: *AIS - EINE ERKLÄRUNG*. https://www.avesmarine.com/blog/ais-explanation/. Version: Aug 2020

[ZS18] ZHAO, Liangbin ; SHI, Guoyou: A Novel Similarity Measure for Clustering Vessel Trajectories Based on Dynamic Time Warping. In: *Journal of Navigation* 72 (2018), Oct, S. 1–17. http://dx.doi.org/10.1017/S0373463318000723. – DOI 10.1017/S0373463318000723

Abbildungsverzeichnis

2.1 Bildschirm eines AIS Radars 2
2.2 Streudiagramm eines Beispielclusterings 5
2.3 Beispiel von Image Embedding 6

3.1 Beispiel von Dynamic Time Warping 8
3.2 Die Berechnungsmatrix mit dem Pfad von DTW 9
3.3 Die Berechnungsmatrix von LCS 10

4.1 Klassendiagramm der Klassen der Implementation 13
4.2 Liniendiagramm zweier Trajektorien 15
4.3 Farbverlauf der Skalierung 16
4.4 Skalierung zweier Trajektorien 16
4.5 Farbverlauf der Geschwindigkeit 17
4.6 Geschwindigkeit zweier Trajektorien 17
4.7 Farbverlauf der Wassertiefe 18
4.8 Wassertiefe zweier Trajektorien als Punkte 18
4.9 Wassertiefe zweier Trajektorien als Karte 19
4.10 Alle Features zweier Trajektorien 20
4.11 Ausgewählte Trawler Trajektorien 21
4.12 Ein Ausschnitt der drawing.log Datei 22

5.1 Streudiagramm des Grundkonstrukts 24
5.2 Liniendiagramme zweier Trajektorien 25
5.3 Streudiagramm der Skalierung 25
5.4 Streudiagramm der Geschwindigkeit 26
5.5 Geschwindigkeit zweier Trajektorien 26
5.6 [Streudiagramm der Wassertiefe als Punkte 27
5.7 Trajektorie 4 mit Wassertiefe als Punkte 27
5.8 Streudiagramm der Wassertiefe als Karte 28
5.9 Liniendiagramm zweier Trajektorien mit Karte 29
5.10 Streudiagramm aller Features und Punkte als Wassertiefe 29
5.11 Streudiagramm aller Features und Karte als Wassertiefe 30

Tabellenverzeichnis

2.1 Ein Ausschnitt aus dem Datensatz 3

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Die eingereichte schriftliche Fassung der Arbeit entspricht der auf dem elektronischen Speichermedium.

Weiterhin versichere ich, dass diese Arbeit noch nicht als Abschlussarbeit an anderer Stelle vorgelegen hat.

Kiel, den 14. Januar 2025

Bahne Thiel-Peters