Universität Paderborn — Fakultät Wirtschaftswissenschaften
Fachgebiet Analytic Information Systems and Business Intelligence
Jun.-Prof. Dr. Artus Krohn-Grimberghe

AIS-BI

Bachelorarbeit

Erkennung körperlicher Aktivitäten mittels Smartphone- und Smartwatch-Sensoren und Machine Learning

Christian Brüggemann

31.03.2017

Betreut von:
Jun.-Prof. Dr. Artus Krohn-Grimberghe

Bachelorarbeit
extern am Fachgebiet Analytic Information Systems and Business Intelligence
Fakultät Wirtschaftswissenschaften
Jun-.Prof. Dr. Artus Krohn-Grimberghe

Institut für Informatik
Fakultät für Elektrotechnik, Informatik und Mathematik
Universität Paderborn

Vorgelegt von:
Christian Brüggemann
Matrikelnummer: 7004878
Salbeiweg 39
33100 Paderborn

am
31.03.2017

Betreut durch:
Jun.-Prof. Dr. Artus Krohn-Grimberghe
Zweitgutachter:
Prof. Dr. Eyke Hüllermeier

Zusammenfassung

Handelsübliche Smartphones und Smartwatches sowie Fitness-Tracker enthalten Sensoren, mit denen sich die körperlichen Aktivitäten des Benutzers aufzeichnen und auswerten lassen. Frühere Forschung widmete sich bereits der Aktivitätenerkennung mittels Smartphones und Smartwatches, jedoch wurden die Daten bisher voneinander getrennt behandelt. Diese Bachelorarbeit widmet sich der Kombination der Datenquellen, um die Vorteile beider Gerätetypen zu vereinen und Modelle mittels maschinellem Lernen zu bilden, die sowohl handorientierte als auch allgemeine Aktivitäten mit hoher Genauigkeit klassifizieren können. Die entwickelte Methode wurde anhand eines Datensatzes getestet, der im Rahmen dieser Arbeit durch ein Experiment mit 10 Personen aufgenommen wurde.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

1.1. Motivation

Smartphones haben im letzten Jahrzehnt an großer Bedeutung gewonnen. Daneben existieren mittlerweile ergänzend dazu sogenannte *Smartwatches* und *Fitness-Tracker*. Smartwatches sind Armbanduhren, die in der Regel drahtlos mit einem Smartphone verbunden sind und Informationen wie beispielsweise Benachrichtigungen am Handgelenk zugänglich machen. Fitness-Tracker besitzen ähnliche Funktionen, zielen allerdings primär darauf ab, die Fitness und Gesundheit des Nutzers zu fördern, indem Daten wie beispielsweise die Schrittanzahl des Nutzers pro Tag gesammelt und grafisch aufbereitet werden. In beiden Geräteformen werden üblicherweise Sensoren verbaut, mit denen sich die Bewegungen des Trägers nachvollziehen lassen.

Mit einigen Fitness-Trackern des Unternehmens *Fitbit* existieren bereits kommerzielle Produkte, die über die reine Sammlung und grafische Aufbereitung von Daten hinausgehen: Die Funktion *SmartTrack* erkennt kontinuierliche Aktivitäten mit hoher Bewegung mit Hilfe von Sensordaten des Trackers teilweise automatisch, ohne dass der Anwender vorher manuell einstellen muss, welcher Aktivität er in den nächsten Minuten nachgehen wird [9]. Dies hat den Vorteil, dass der Nutzer sich nicht daran erinnern muss, im Fitness-Tracker die richtige Aktivität einzustellen, um kategorisierte Statistiken zu erhalten.

SmartTrack unterstützt die folgenden Aktivitäten: Gehen, Laufen, Fahrradfahren, Schwimmen und Training mit einem Crosstrainer, sowie zwei allgemeine Kategorien „Sport“ (Fußball, Basketball, Tennis, etc.) und „aerobes Training“ (Zumba, Tanzen).

Es existieren weitere mögliche Anwendungsgebiete der automatisierten Aktivitätenerkennung. Für Smartphone-Betriebssysteme könnte das Wissen, dass der Anwender gerade Sport treibt, interessant sein, um eingehende Anrufe eines nicht als wichtig markierten Kontaktes zu unterdrücken. Des Weiteren könnte das Forschungsgebiet der „Transportation Mode Recognition“ von solchen Methoden profitieren: Soll erkannt werden, mit welchem Verkehrsmittel sich der Nutzer gerade fortbewegt, könnte neben dem Parameter der Geschwindigkeit ebenfalls von Interesse ein, ob mit Hilfe der Methode die Aktivität „Fahrradfahren“ erkannt wird oder nicht. So ließe sich die Fortbewegung mittels eines Mofas von der Fortbewegung mittels eines Fahrrads unterscheiden, was insbesondere für Dienste wie „Google Now“ nützlich sein könnte. Diese dienen dem Nutzer als persönlicher Assistent und warnen ihn beispielsweise vor Stau auf einer häufig befahrenen Strecke. Eine solche Warnung könnte entfallen, wenn festgestellt wurde, dass der Nutzer die Strecke nicht mit einem Motorroller, sondern mit einem Fahrrad bewältigt und somit Radwege befahren darf.

In der Literatur ist [29] hervorzuheben. Die Autoren vergleichen in ihrem Paper die Genauigkeit der Aktivitätenerkennung eines Smartphones mit der einer Smartwatch und kommen zu dem Schluss, dass die Güte der jeweiligen Erkennung insbesondere von der Aktivität selbst abhängig ist. Es liegt auf der Hand, dass nur mit Hilfe eines Smartphones beispielsweise eine

Unterscheidung zwischen „Zähneputzen“ und „Stehen“ schwer möglich ist, während analog dazu nur mit Hilfe einer Smartwatch die Unterscheidung zwischen „Gehen“ und „Fußball schießen“ ebenfalls herausfordernd ist. Naheliegend ist daher, eine Kombination beider Datenquellen einzusetzen, um die durchschnittliche Erkennungsrate zu verbessern, ohne eine Beschränkung der erkennbaren Aktivitäten einzuführen. Im folgenden Abschnitt werden die Ziele dieser Arbeit genauer definiert.

1.2. Ziele der Arbeit

Evaluiert werden soll ein zu entwickelndes Verfahren, das mit Methoden des *maschinellen Lernens* (siehe Definition 1) und eben jenen gesammelten Daten feststellt, welcher Aktivität der Träger der Geräte in bestimmten Zeitintervallen nachgegangen ist. Hierzu wird zunächst eine Software benötigt, welche die synchrone Aufzeichnung von Sensordaten eines Smartphones und zusätzlich eines Fitness-Trackers oder einer Smartwatch ermöglicht.

Es ergibt sich insbesondere die Frage, inwiefern sowohl personalisierte, das heißt nutzerspezifische, als auch unpersonalisierte Modelle durch die Hinzunahme einer weiteren Datenquelle genauer werden.

Um eine Evaluation zu ermöglichen, wird ein Beispieldatensatz benötigt, der durch ein Experiment mit 10 Probanden aufgebaut wird. Orientiert ist diese Zahl an der Anzahl der Probanden in [29], an dessen Experiment 17 Personen teilgenommen haben. Im Experiment sollen diese voneinander unabhängig mehreren definierten Aktivitäten nachgehen, während parallel dazu Sensordaten mithilfe der entwickelten Software aufgezeichnet werden. Um die Vergleichbarkeit mit den Ergebnissen aus [29] zu gewährleisten, werden die Probanden in diesem Experiment denselben Aktivitäten nachgehen.

Als Fitness-Tracker wurde das *Microsoft Band 2* ausgewählt, da dieser vom Lehrstuhl für diese Arbeit zur Verfügung gestellt wurde, im Vergleich zur privat vorhandenen Smartwatch *Pebble Time* mehr Sensoren besitzt und letztere während der Durchführung des Experiments nach einem erzwungenen Software-Update falsche Zeitstempel für Sensordaten lieferte.

1.3. Wichtige Ergebnisse

Hervorzuheben sind die Erkenntnisse, dass die Genauigkeit persönlicher Modelle mithilfe der Kombination der in Abschnitt 2.1 vorgestellten Sensoren auf 99.4% gesteigert werden konnte. Im Vergleich zu Modellen, die nur auf Daten des Beschleunigungssensors eines Fitness-Trackers basieren, beträgt die Steigerung damit 7.8 Prozentpunkte. Unpersönliche Modelle hingegen konnten durch diese Technik um 3.3 Prozentpunkte auf 78.5% verbessert werden, wobei gleichzeitig die Stabilität insofern gesteigert wurde, dass die schlechteste Genauigkeit für einen Teilnehmer des Experiments mit einem unpersönlichen Modell nicht unter 64% lag. Um die Genauigkeit unpersönlicher Modelle weiter zu verbessern, empfiehlt es sich, nur klar voneinander abgrenzbare Aktivitäten zu klassifizieren und beispielsweise mehrere Essaktivitäten zu vermeiden.

Des Weiteren konnte festgestellt werden, dass auch niedrige, einstellige Sensorabtastraten in Hz bei einem niedrigeren Energieverbrauch noch gute Ergebnisse liefern können und

die Variation hinsichtlich der Ausrichtung der Geräte nur einen geringen Einfluss auf die Genauigkeiten der Modelle hat.

1.4. Struktur dieser Arbeit

Kapitel 2 erläutert die verwendeten Sensoren sowie wichtige Begriffe des ML als Grundlagen dieser Arbeit. Im darauffolgenden Kapitel 3 werden verwandte Arbeiten aufgezählt und historisch eingeordnet, da insbesondere die Entwicklung mobiler Technologie diverse Fortschritte im Bereich der Aktivitätenerkennung erst ermöglicht hat. Anschließend daran wird in Kapitel 4 der Aufbau des Experiments beschrieben, das im Rahmen dieser Bachelorarbeit durchgeführt wurde, um die benötigten Daten zu sammeln. Die Verarbeitung der gewonnenen Daten wird in Kapitel 5 erläutert. Die aus der Anwendung der Methode resultierenden Modelle werden in Kapitel 6 evaluiert, wobei unter anderem diverse Eigenschaften des aufgenommenen Datensatzes variiert werden, um Erkenntnisse über den Wert der Sensoren für die Aktivitätenerkennung zu erhalten. Ein abschließendes Fazit der Arbeit erfolgt in Kapitel 7.

2. Grundlagen

2.1. Erläuterung der verwendeten Sensoren

Während in der Einleitung nur generisch von „Sensordaten“ die Rede war, werden diese nun konkretisiert. Zur Aufnahme werden das Android-Smartphone OnePlus 3 sowie der Fitness-Tracker Microsoft Band 2 verwendet, der mit dem Smartphone via Bluetooth verbunden ist. Beide Geräte besitzen einen triaxialen Beschleunigungssensor sowie ein triaxiales Gyroskop. Des Weiteren besitzt das Band unter anderem einen Sensor, der den elektrischen Widerstand der Haut des Trägers misst sowie einen Hauttemperatursensor.

2.1.1. Beschleunigungssensor

Ein Beschleunigungssensor, auch *Accelerometer* genannt, misst die echte Beschleunigung eines Objektes bezüglich der Senkrechte je physikalischer Achse (x, y und z). Dies beinhaltet auch die Beschleunigung, die von der Erdgravitation ausgeht: Liegt das Objekt beispielsweise auf dem Boden, erfährt es auf der im rechten Winkel zum Boden stehenden Achse eine Beschleunigung von $g \approx 9.81\ m/s^2$ [13, 27]. Über diesen Sensor kann demnach verfolgt werden, wie der Träger sich im Raum bewegt. Abbildung 2.1 illustriert die Sensordaten eines unbewegten Smartphones, dessen Bildschirm parallel zum Boden gehalten wird.
Beide Geräte verfügen über einen solchen Sensor.

2.1.2. Gyroskop

Ein Gyroskop, normalerweise Gyrometer genannt, misst die Rotationsgeschwindigkeit je physikalischer Achse [13]. Demnach bedeutet der Messwert $(x = 0, y = 0, z = 0)$, dass das

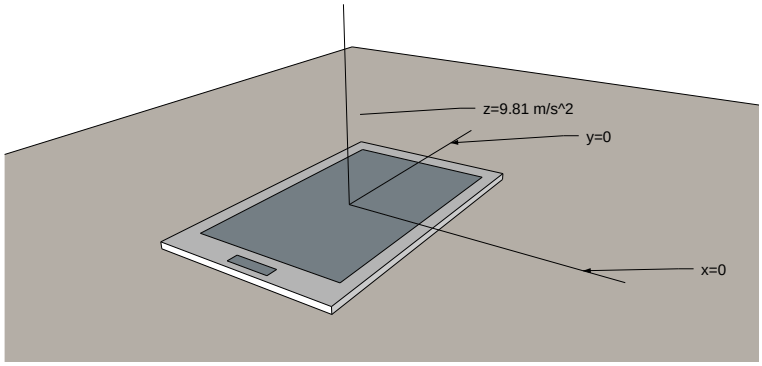


Abbildung 2.1.: Illustration des Beschleunigungssensors

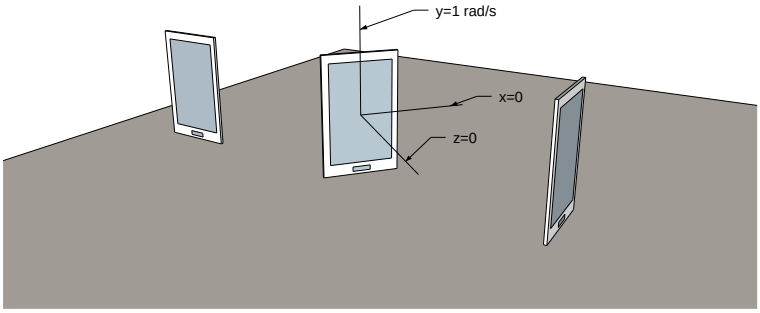


Abbildung 2.2.: Illustration des Gyroskops. Die Achsenbeschriftungen geben die Rotationsgeschwindigkeit um die jeweilige Achse an. Von links nach rechts wird dasselbe Gerät bei fortschreitender Zeit gezeigt.

Objekt relativ zur Erde nicht bewegt wird, während $(x = 0, y = 1, z = 0)$ bedeutet, dass das Objekt um die y-Achse gedreht wird, wie es beispielhaft in Abbildung 2.2 von links nach rechts gezeigt wird.
Beide Geräte verfügen über einen solchen Sensor.

2.1.3. Hautwiderstandssensor

Je nach aktueller körperlicher Betätigung ändert sich die elektrische Leitfähigkeit der Hautoberfläche durch Schweiß. Der Hautwiderstandssensor misst den elektrischen Widerstand der Haut, der sich umgekehrt proportional zur Leitfähigkeit verhält. Somit können Aussagen über die physische oder psychische Belastung des Trägers getroffen werden. Dieser Sensor ist nur im Band integriert, das daher für diesen die einzige Datenquelle ist.

2.1.4. Hauttemperatursensor

Der Hauttemperatursensor misst die Temperatur der Haut des Nutzers, die an der Unterseite des Band anliegt. Leider reagierte dieser Sensor in einem Test nur langsam auf Temperaturveränderungen: Nachdem der Tracker abgelegt wurde, übermittelte dieser den Temperaturabfall erst mehrere Sekunden später. Aus diesem Grund wurde auf die Aufzeichnung dieser Daten verzichtet.

2.1.5. Laufgeschwindigkeitssensor

Der Laufgeschwindigkeitssensor ist ein virtueller Sensor, der auf Basis des elektronischen Schrittzählers im Microsoft Band 2 generiert wird. Er gibt die geschätzte Laufgeschwindigkeit des Trägers in cm/s an.

2.2. Wichtige Begriffe

In den folgenden Kapiteln wird von einem grundlegenden Verständnis von ML-Klassifikation ausgegangen.

Definition 1 (ML-Klassifikation)**.** *Gegeben sei ein Datensatz $D \ni (x_1,...,x_n,y)$ mit $|D| = m$ Instanzen. Dann sind $x = (x_1,...,x_n)$ die Input-Features und y das Target. Im Falle eines Klassifikationsproblems ist y nominal und beschreibt in Form eines Texts eine Klasse. Gesucht ist nun eine Hypothesenfunktion $h(x) = \hat{y}$, die zu gegebenen Features den wahrscheinlichsten Wert des Targets bestimmt [23].*

Neben Definition 1 sind für das Verständnis dieser Arbeit die folgenden Begriffe wichtig:

1. *(Überwachtes) Lernen, bzw. Training* und *Modelle.* Einem ML-Algorithmus werden Trainingsdaten $T_1 \subset D$ gezeigt, zu denen das Target bekannt ist. Aus diesem Prozess ergibt sich ein gelerntes *Modell*.

   Weitere Informationen befinden sich in Hastie, Kapitel 2.1 [11].

2. *Overfitting und Train-Test-Split.* Um die Genauigkeit eines Modells zu evaluieren, müssen mit Hilfe dessen Testdaten $T_2 \subset D$ klassifiziert werden, zu denen das Target ebenfalls bekannt ist. Ist $T_1 \cap T_2 \neq \emptyset$, so besteht das Risiko, dass der ML-Algorithmus sich den Trainingsdaten zu sehr angepasst hat, daher auch im Test mit T_2 genau ist und auf weiteren, noch unbekannten Daten eine schlechte Genauigkeit erzielt (*Overfitting*).

   Weitere Informationen befinden sich in Hastie, Kapitel 7.2 [11].

3. *Kreuzvalidierung.* Bei einer k-fachen Kreuzvalidierung werden k Paare T_1, T_2 mit $T_1 \cap T_2 = \emptyset$ erstellt, sodass jedes Datum einmal in T_2, dann jedoch nicht in T_1 enthalten ist. Anschließend wird ein ML-Klassifikationsalgorithmus mit jedem T_1 trainiert und dem dazugehörigen T_2 ausgewertet.

   Weitere Informationen befinden sich in Hastie, Kapitel 7.10 [11].

4. *Hyperparameter.* Dabei handelt es sich um Parameter eines ML-Algorithmus, die dieser nicht eigenständig optimieren kann.

Vor der Entwicklung eines Verfahrens zur Erkennung körperlicher Aktivitäten mittels Smartphone- und Smartwatch-Sensoren und Machine Learning steht die Untersuchung verwandter Arbeiten, die im nächsten Kapitel folgt.

3. Verwandte Arbeiten

Dieses Kapitel behandelt verwandte Arbeiten, die bis in das Jahr 1998 zurückreichen. Im November 1998 veröffentlichten Schmidt et al. im Rahmen des *Technology for Enabling Awareness (TEA)*-Projektes ein Paper über *Context Awareness in Ultra-Mobile Computing*, in dem sie darauf hinwiesen, dass Kontext mehr beinhaltet als den Ort des Geschehens, auf den sich vorige Werke konzentrierten [26]. In diesem Paper schlugen sie neben der Verwendung expliziter Regeln zur Erkennung diverser Aktivitäten zusätzlich auch die Verwendung von Methoden künstlicher Intelligenz vor.

Parallel dazu erkannte Ashbrook 1999 ebenfalls die Bedeutung der Kontexterkennung als wichtige Komponente tragbarer Computeranwendungen [2]. Ein damals kommerziell verfügbares Produkt namens *Twiddler*, eine mit nur einer Hand bedienbare Tastatur, integrierte zwei Sensoren, welche die Ausrichtung des Gerätes feststellen konnten. Mit der Motivation, automatisch eine To-Do-Liste zu öffnen, wenn der Nutzer des Gerätes sich zu seinem nächsten Termin bewegt, untersuchte Ashbrook die Möglichkeit, eben diese Tätigkeit mit Hilfe der genannten Sensoren automatisch erkennen zu lassen. Er schlug mehrere solche Methoden vor, die allerdings allesamt keinen Gebrauch von Machine Learning machten und nur schwer auf andere Aktivitäten übertragbar waren.

Ebenfalls im Jahr 1999 äußerten Farringdon et al. Kritik an Mobilgeräten wie Handys und PDAs, die ihre Besitzer auch in unangemessenen Situationen visuell und akustisch über eingehende Nachrichten benachrichtigen [8]. Um dieses Problem anzugehen entwickelten sie ein an der Hüfte tragbares Gerät, das zwei einachsige Accelerometer integrierte, sowie eine Jacke, die die Dehnung des Materials messen konnte. Auch hier wurde Machine Learning nicht eingesetzt. Stattdessen wurde für die Erkennung der Aktivitäten Sitzen, Stehen, Liegen, Gehen und Laufen ein Algorithmus entwickelt, der Aspekte wie die Durchschnittswerte der Sensoren betrachtet und mit Schwellwerten vergleicht.

Das TEA-Projekt wurde über mehrere Jahre fortgeführt. Nach den Publikationen von Ashbrook und Farringdon et al. wurde im Rahmen dieses Projektes 2000 ein Paper von Van Learhoven und Cakmakci veröffentlicht, das unter anderem die Arbeit von Ashbrook anerkannte, jedoch darauf hinwies, dass Kontexterkennung adaptiv sein muss und sich dafür Methoden des maschinellen Lernens anbieten [28]. Als mindestens eines der ersten Werke zu diesem Thema setzen die Autoren mehrere Sensortypen ein und verwenden neben Beschleunigungssensoren auch Infrarot-, Temperatur-, Kohlenstoffmonoxid-, Berührungs-, Luftdruck- und Lichtsensoren sowie Mikrofone, die in einem am Oberschenkel tragbaren Gerät integriert wurden. Diese Fülle von Daten verursachte zu diesem Zeitpunkt noch Probleme hinsichtlich des Rechenaufwands, weshalb die Autoren diverse Vorverarbeitungstechniken wie beispielsweise eine Fouriertransformation einsetzten, um die Datenmenge zu reduzieren. Anschließend verwendeten sie eine *Kohonne Self-Organizing Map (KSOM)* als neuronenbasiertes Clusteringverfahren, in dem verschiedene Eingabewerte nach dem Trainingsprozess verschiedene Neuronenareale aktivieren. Dies ermöglichte eine Visualisierung des Clusterings

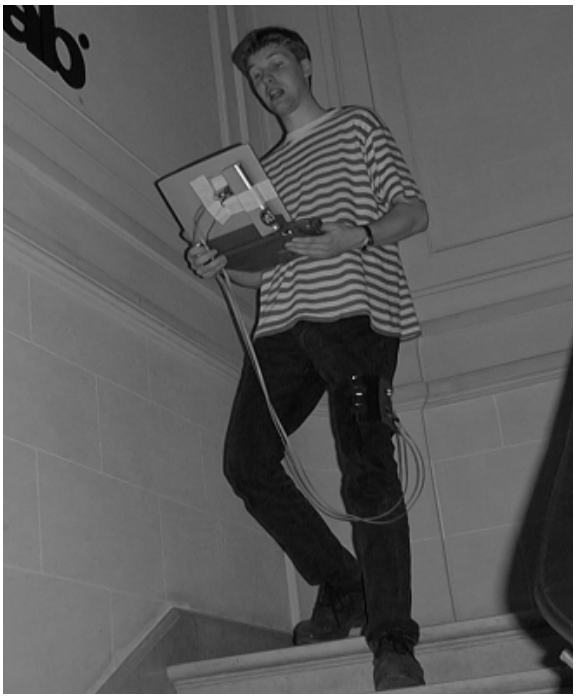


Abbildung 3.1.: Der Aufnahmeprozess von Van Learhoven und Cakmakci [28]

und ließ die Autoren darauf schließen, dass eine Erkennung von Aktivitäten mit Hilfe von Machine Learning grundsätzlich möglich ist. Auf Basis der KSOM arbeitete anschließend ein *k*NN-Verfahren: Für die durch eine Eingabe aktivierten Neuronen wurden die *k* nächsten Nachbarn ermittelt, zu denen bekannt war, welche Aktivität sie aktiviert. Die unter diesen Nachbarn am häufigsten vorkommende Aktivität wurde das Ergebnis des Klassifikators. Dieses Verfahren, das auf einem Notebook in Echtzeit arbeitete, lieferte für die Aktivitäten Sitzen, Stehen, Gehen, Laufen und Fahrradfahren gute Ergebnisse, nur das Treppensteigen sorgte für Probleme. Abbildung 3.1 zeigt die Einschränkungen des Experiments durch die beschränkte Rechenleistung und Speicherkapazität der zum Zeitpunkt der Arbeit verfügbaren mobilen Technologie.

Nicht nur im Sinne der *Context Awareness* gab es Entwicklungen im Bereich der Aktivitätenerkennung. Auch für medizinische Zwecke hat diese Technologie Relevanz, beispielsweise bei Rehabilitationstherapien. 2001 veröffentlichten Bussmann et al. ein Paper über den sogenannten *Activity Monitor*, der für diesen Zweck entwickelt wurde [6]. Um die Aktivitäten Liegen, Sitzen, Stehen, Gehen, Laufen, Treppensteigen, Fahrradfahren, Rollstuhlfahren und Übergänge dazwischen erkennen zu können, wurde ein tragbares Gerät entwickelt, das Sensordaten aufzeichnet. An mehreren Körperstellen wurden Sensoren befestigt, die kabelgebunden Daten an einen Rekorder übermittelten, der in einem Beutel an der Hüfte des Trägers angebracht wurde. Aus den Rohdaten wurden Features extrahiert, die allerdings nicht für das Training eines ML-Algorithmus eingesetzt wurden, sondern für die manuelle Erstellung einer Datenbank zur Erkennung der Aktivität. Dafür wurden für die extrahierten Features pro Aktivität Minima und Maxima definiert, sodass die summierte Distanz der Eingabefeatures von

diesen Intervallen zur Klassifikation berechnet werden konnte. Dieser manuelle Prozess ermöglichte in vier Studien Übereinstimmungen der Klassifikation mit den tatsächlichen Aktivitäten von $89\%, 93\%, 81\%$ und 90%. Eine große Schwäche des Gesamtkonzepts war die Größe des Activity Monitors, durch die Aktivitäten beeinflusst oder sogar vollständig verhindert wurden. Ein weiteres Problem waren die mit 10000 USD angegebenen Kosten des Gerätes. Dank der heutigen Technik wurden diese Probleme weitgehend eliminiert, jedoch verbleibt ein ethisches Problem, das die Autoren erkannten: Der Monitor könnte als *Big Brother* aufgefasst werden, der die Privatsphäre des Benutzers einschränkt. Insbesondere im sensiblen medizinischen Kontext trifft dies zu.

Bao und Intille prüften im Jahre 2004 in einer Studie mit 20 Teilnehmern und Aktivitäten, ob Aktivitätenerkennung auch außerhalb von Laborbedingungen praktikabel ist, da sie vermuteten, dass sich Menschen im Labor anders verhalten als im alltäglichen Leben. Des Weiteren wurden zur größeren Bewegungsfreiheit diverse am Körper installierte Sensoren nicht mit einem zentralen Rekorder verbunden, sodass jeder Sensor separate Aufnahmen erzeugen musste. Dies warf wie in dieser Arbeit die Frage auf, inwiefern Abweichungen der Uhren der Aufnahmegeräte zum Problem werden könnten, weshalb man sich zu einer Synchronisierung zum Start und Ende der Aufnahme entschlossen hat. Um möglichst realitätsnahe Aufnahmen zu erzeugen, entwickelten Bao und Intille einen Parcours mit Aufgabenstellungen, die die aufzunehmenden Aktivitäten beinhalteten, jedoch nicht als Hauptziel hatten. Die Teilnehmer des Experiments wurden dabei nicht beobachtet und mussten selbst notieren, wann sie eine Aufgabe angefangen und abgeschlossen hatten. Einige Aktivitäten wurden außerhalb eines Labors aufgenommen. Auf Basis dieser Daten wurden mit Hilfe von ML-Klassifikationsalgorithmen Modelle trainiert und ausgewertet. Mit einer Genauigkeitsrate von etwa 85% waren die Ergebnisse für Modelle, die keine nutzerspezifischen Informationen beinhalteten, vergleichbar mit Ergebnissen, die in anderen Werken unter Laborbedingungen erzeugt wurden. Dies deutet darauf hin, dass ein aufwendiger Parcours nicht unbedingt notwendig ist, um aussagekräftige Ergebnisse zu erhalten. Eine weitere interessante Erkenntnis von Bao und Intille war, dass ein Beschleunigungssensor am Oberschenkel die größte Aussagekraft hatte und dass ein Beschleunigungssensor am dominanten Arm des Trägers nützlicher war als am nichtdominanten Arm. Die zweitgrößte Aussagekraft hatte ein Sensor an der Hüfte des Trägers, woraus die Autoren schlossen, dass ein am Handy angebrachter Sensor ähnliche Ergebnisse erzielen könnte. Dies deutet auf gute Voraussetzungen für meine Methode hin, in der Daten von einem Smartphone in der Hosentasche und von einem Fitness-Tracker am dominanten Arm aufgenommen werden.

2005 untersuchten Ravi et al. mit einem via Bluetooth verbundenen Beschleunigungssensor unter anderem, welche Features im Kontext der Aktivitätenerkennung sinnvoll und welche Aktivitäten besonders schwer zu erkennen sind [25]. Eines der getesteten Features war die sogenannte *Energie* als Summe der Komponenten, die aus einer Fouriertransformation der sequentiellen Daten hervorging. Da eine Fouriertransformation diese in ein Frequenzspektrum zerlegt, vermuteten die Autoren in diesem Feature die Periodizität der Aktivitäten widerspiegeln zu können, jedoch erwies sich dieses Feature nicht als signifikant. Wie Bao und Intille stellten auch die Autoren dieser Arbeit fest, dass ein Beschleunigungssensor auf Hüfthöhe Aktivitäten gut erkennen kann, jedoch bei handorientierten Aktivitäten Schwächen zeigt.

Thematisch anknüpfend an die bereits 2004 gewonnenen Erkenntnisse von Bao und Intille veröffentlichten Kwapisz, Weiss und Moore 2011 ein Paper, das erfolgreiche ML-basierte Aktivitätenerkennung auf einer Datenbasis demonstrierte, die exklusiv mit Hilfe von Smartphone-

Beschleunigungsensoren gewonnen wurde [17]. Die Arbeit der Autoren lieferte die Grundlage für den in Abschnitt 5.2 dieser Arbeit beschriebenen Transformationsprozess. 2012 untersuchten Weiss und Lockhart auf Basis dieser Methode, inwiefern die Personalisierung von Modellen durch nutzerspezifische Aufnahmen diese verbessert und kamen zu dem Schluss, dass eine Personalisierung die Genauigkeit bedeutend verbessert [30]. 2016 verwendeten Weiss et al. statt eines Smartphones erfolgreich eine Smartwatch und demonstrierten insbesondere für handorientierte Aktivitäten starke Verbesserungen der Klassifikationsgenauigkeit [29].

Die Kombination mehrerer Smartphone-Sensoren wurde 2012 erstmals von Suarez et al. durchgeführt und demonstrierte, dass damit eine Verbesserung der Genauigkeit um 10% und mehr möglich ist [7].

Bei der weiteren Recherche fanden sich keine Arbeiten, die sich mit der Kombination der Daten aus Smartphone und Fitness-Tracker befassen. Somit ist davon auszugehen, dass die vorliegende Arbeit die erste ist, in der die Daten mehrerer Sensoren eines Smartphones mit den Daten mehrerer Sensoren eines Fitness-Trackers kombiniert werden. Bevor ein Verfahren entwickelt und evaluiert werden kann, muss jedoch zunächst ein Datensatz dafür erstellt werden. Damit beschäftigt sich das folgende Kapitel.

4. Experiment

In diesem Kapitel werden Aufbau und Durchführung des Experiments erläutert, durch das der Datensatz zusammengestellt wurde.

Insgesamt haben 10 Teilnehmer am Experiment teilgenommen und die in Abbildung 4.1 achtzehn gelisteten Aktivitäten durchgeführt. Weiss et al. nahmen jeweils zwei Minuten pro Teilnehmer und Aktivität auf [29], mussten allerdings zum Anfang und Ende jeder Aufnahme je zehn Sekunden abschneiden, um Ausreißer zu entfernen. Aus diesem Grund betrug die Dauer jeder Aufnahme in diesem Experiment drei Minuten, wodurch der gesamte Aufnahmeprozess pro Person rund zwei Stunden dauerte. Vor dem Experiment wurden Einverständniserklärungen der Teilnehmer eingeholt.

Die Aufnahme erfolgte mit dem Fitness-Tracker Microsoft Band 2 und dem Smartphone OnePlus 3, auf dem das Betriebssystem Android 6 installiert war. Der Fitness-Tracker wurde am Handgelenk des Teilnehmers befestigt, während das Smartphone in in einer frontseitigen Hosentasche platziert wurde, wie in [29] jeweils auf der dominanten Seite des Teilnehmers. Dies war notwendig, da insbesondere Aktivitäten wie das Dribbeln eines Basketballs mit der dominanten Hand durchgeführt werden. Die Ausrichtung des Smartphones in der Hosentasche selbst war im Experiment nicht festgelegt.

Auf dem Smartphone wurde eine eigens für das Experiment entwickelte Anwendung ausgeführt, in der zunächst der Name des Teilnehmers eingegeben wurde. Sowohl auf dem Smartphone selbst als auch per drahtloser Fernsteuerung wurde die durchzuführende Aktivität eingestellt. Gestartet und gestoppt wurde die Aufnahme anschließend per Fernsteuerung, um Ausreißer am Anfang und Ende der Aufnahme zu vermeiden, obgleich sicherheitshalber trotzdem jeweils zehn Sekunden abgeschnitten wurden.

4.1. Beschreibung der Aktivitäten

Im Folgenden werden die einzelnen Aktivitäten, die von den Teilnehmern ausgeführt wurden, genauer beschrieben. Um eine Vergleichbarkeit mit [29] zu ermöglichen, handelt es sich mangels detaillierter Beschreibungen zumindest um ähnliche Aktivitäten, die dieselben Bezeichnungen haben.

4.1.1. Allgemeine Aktivitäten (nicht handorientiert)

Gehen

Der Teilnehmer bewegt sich im Schrittempo auf einem Gehweg.

Joggen

Der Teilnehmer joggt auf einem Gehweg. Das Tempo variiert je nach sportlicher Verfassung des Teilnehmers.

Treppensteigen

Der Teilnehmer bewegt sich abwechselnd eine Treppe hoch und herunter. Steigung und Länge sind variabel.

Sitzen

Der Teilnehmer sitzt auf einem Stuhl und versucht, sich dabei nicht unruhig zu verhalten.

Stehen

Der Teilnehmer steht und versucht, sich dabei nicht unruhig zu verhalten.

Fußball schießen

Der Teilnehmer schießt einen Fußball wiederholt zu einem wenige Meter entfernten Mitspieler und versucht, Sprints zu vermeiden.

4.1.2. Allgemeine Aktivitäten (handorientiert)

Basketball dribbeln

Der Teilnehmer wirft einen Basketball wiederholt auf den Boden und versucht, dabei möglichst an einer Stelle stehen zu bleiben.

Mit einem Tennisball Fangen spielen

Zwei Personen werfen sich abwechselnd einen Tennisball zu und versuchen dabei, möglichst an einer Stelle stehen zu bleiben.

Auf einer Tastatur tippen

Der Teilnehmer tippt sitzend seinen Gewohnheiten nach einen Text an einer beliebigen Computertastatur ab. Mindestens grobe Fehler sollten korrigiert werden. Um Leseschwierigkeiten aus dem Weg zu gehen, handelt es sich bei dem Text um ein Diktat für Siebtklässler.

Auf Papier schreiben

Der Teilnehmer schreibt sitzend seinen Gewohnheiten nach denselben Text wie mit der Tastatur mit einem Kugelschreiber auf ein Blatt Papier im Format DIN A4 ab.

Klatschen

Der Teilnehmer klatscht im Takt sitzend zum Lied „Viva la Vida“ von Coldplay.

Zähneputzen

Der Teilnehmer putzt sich stehend mit einer nicht-elektrischen Handzahnbürste die Zähne.

Kleidung falten

Der Teilnehmer faltet stehend der Reihe nach Kleidung auf einem Tisch.

4.1.3. Essaktivitäten (handorientiert)

Spaghetti essen

Der Teilnehmer isst Spaghetti mit einer beliebigen Soße.

Suppe essen

Der Teilnehmer isst eine beliebige Suppe.

Brot essen

Der Teilnehmer isst ein belegtes Brot.

Chips essen

Der Teilnehmer isst Chips aus einer handelsüblichen Tüte.

Aus einer Tasse oder einem Glas trinken

Der Teilnehmer trinkt wiederholt aus einer Tasse oder einem Glas.

4.2. Überwachung der Aufnahme

Zur Qualitätssicherung der Daten wurde die Aufnahme der Aktivitäten überwacht. Den Teilnehmern wurde größtmögliche Freiheit bei der Ausführung gegeben, weshalb lediglich auf starke Abweichungen von der Aufgabenstellung hingewiesen wurde, die über einen Zeitraum von über 10 Sekunden hinweg begangen wurden. Als Beispiele seien hierfür das beabsichtigte Werfen des Tennisballs auf den Boden und das Gestikulieren in der Luft während des Tippens auf der Tastatur genannt, wobei das eigentliche Tippen auf dieser eingestellt wurde.

Um eine sonstige Beeinflussung der Bewegungsabläufe zu vermeiden, wurde auf einen festen Aufnahmeort in einem Labor verzichtet, so wie es auch Bao & Intille zumindest für einige Aktivitäten vermieden haben [3]. Stattdessen wurden die Experimente in einem freizeitlichen Kontext und in unterschiedlichen Umgebungen durchgeführt, sodass sich die Teilnehmer möglichst wie in ihrem sonstigen Alltag verhalten würden. In keinem Fall wurde eine Aufnahme, die schon über 10 Sekunden lief, aufgrund von Fehlverhalten abgebrochen oder nicht weiterverwendet.

Teilnehmer	Geschlecht	Alter	Beruf	Größe	Dom. Seite
1	w	47	Zahnmed. Fachangestellte	171 cm	R
2	m	22	Student	176 cm	R
3	m	20	Konstruktionsmechaniker	187 cm	R
4	m	16	Schüler	182 cm	R
5	w	18	Med. Fachangestellte	163 cm	R
6	m	21	Student	181 cm	R
7	m	50	Ingenieur	183 cm	R
8	m	22	Student	183 cm	L
9	w	19	Studentin	163 cm	R
10	m	50	Kraftfahrer	178 cm	R

Abbildung 4.1.: Attribute der Teilnehmer

4.3. Auswahl der Teilnehmer

Bei der Auswahl der Teilnehmer wurde darauf geachtet, dass die Gruppe nicht übermäßig homogen wurde. Wie Tabelle 4.1 zeigt, reicht die Altersspanne zum Zeitpunkt des Experiments von 16 bis 50, wobei der maximale Körpergrößenunterschied 24 cm beträgt. Unter den 10 Teilnehmern sind drei weiblich und die verschiedenen Berufsstände zeigen die unterschiedlichen Milieuzugehörigkeiten der Personen. Des Weiteren ist einer der 10 Teilnehmer Linkshänder und trug die Geräte somit auf seiner linken Seite. Dieser Anteil entspricht schätzungsweise dem Anteil von Linkshändern unter den seit 1940 geborenen Personen [19]. Teilnehmer 2 ist der Autor dieser Bachelorarbeit.

Da nun ein Datensatz zum Training und zur Evaluation besteht, kann im folgenden Kapitel mit der Entwicklung der Methode fortgefahren werden.

5. Methode

Dieses Kapitel erläutert die Art und Weise, mit der aus den im Experiment gewonnenen Daten Modelle trainiert wurden, welche die ausgeführte Aktivität automatisch erkennen können. Der erste Schritt ist die Aufnahme der Rohdaten, die durch eine in Abschnitt 5.1 beschriebene Android-App realisiert wurde. Die gespeicherten Rohdaten sind noch nicht für einen herkömmlichen ML-Klassifikationsalgorithmus verwertbar, weshalb eine in Abschnitt 5.2 beschriebene Transformation sowie eine anschließende Nachbearbeitung wie in Abschnitt 5.3 erläutert ausgeführt wird. Als dritter und letzter Schritt erfolgt die Ausführung diverser ML-Klassifikationsalgorithmen, die in Abschnitt 5.4 erläutert werden.

5.1. Implementierung der Aufzeichnungssoftware

Die Aufzeichnung der Daten erforderte die Entwicklung einer neuen Software. Mit dem *Actitracker* besteht zwar bereits eine Software von Lockhart et al. [18], jedoch unterstützt diese die synchronisierte Aufnahme mehrerer Sensoren sowie das Microsoft Band 2 generell nicht. Wichtige Anforderungen an die Aufzeichnungssoftware waren in dieser Arbeit eine hohe Zuverlässigkeit, Fernsteuerbarkeit und das automatische Hochladen der Daten auf einen Datenspeicher im Anschluss an die Aufnahme. Teilnehmer des Experimentes sollten mit der Anwendung möglichst wenig in Berührung kommen.

Zunächst wird im Folgenden die Struktur der Messdaten definiert, woraufhin der Aufbau der Aufzeichnungssoftware erläutert wird.

5.1.1. Definition der Messdaten

Definition 2. *Ein Reading besteht aus einem Unix-Zeitstempel in Millisekunden, einer Gerätequelle, einem Sensortyp und den Werten des Sensors gruppiert nach Komponente. Ein Beispiel ist* $(1000, \mathit{Band}, \mathit{Gyroscope}, \{x \to 0.0, y \to 1.0, z \to 1.0\})$.

5.1.2. Struktur der Aufzeichnungssoftware

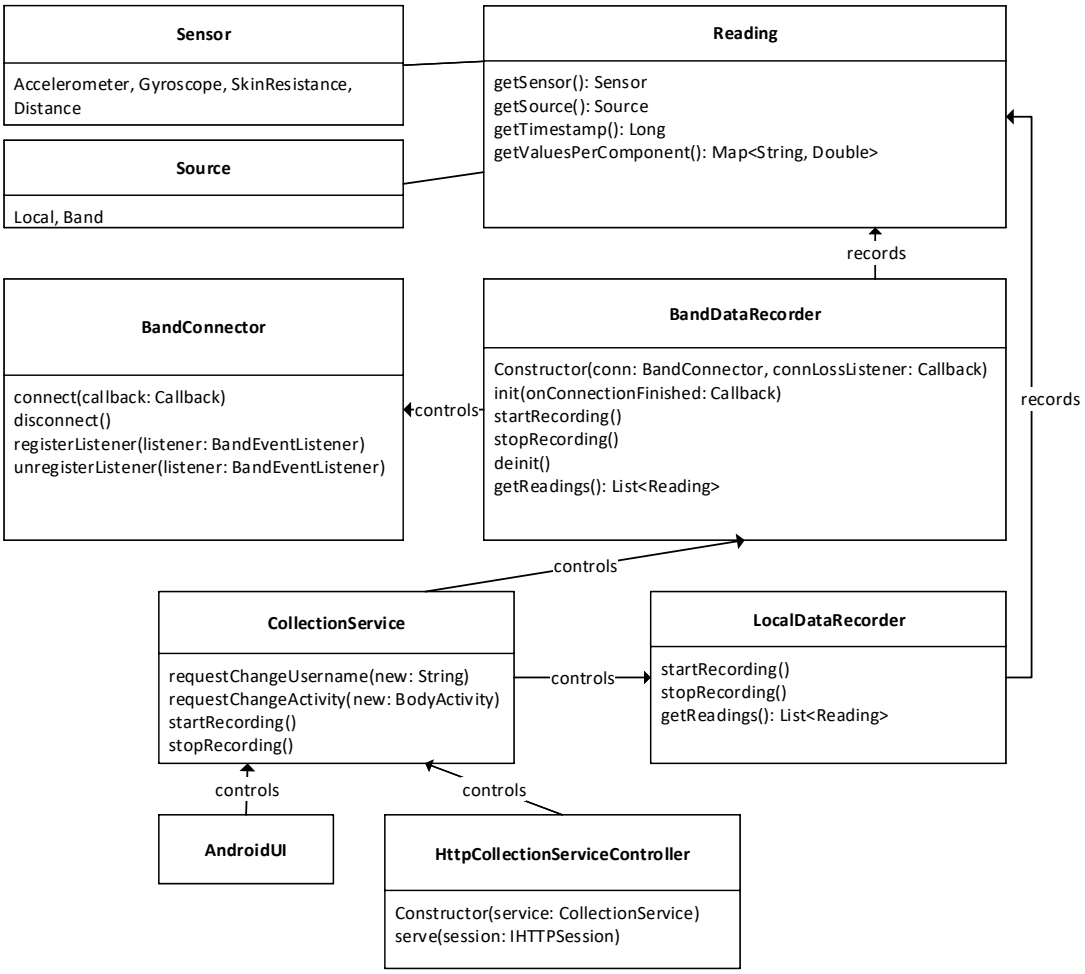


Abbildung 5.1.: Struktur der Aufzeichnungssoftware

Die Aufzeichnungssoftware wurde als Anwendung für das Android-Betriebssystem implementiert, da für diese Plattform auch ein *Software Development Kit (SDK)* für das Microsoft Band 2 existiert.

Abbildung 5.1 bietet einen Überblick über die Struktur der Aufzeichnungssoftware. Im Zentrum steht die Klasse *CollectionService*, die für die robuste Sammlung der Daten verantwortlich ist und daher als persistenter Android-*Service* implementiert ist, der im Gegensatz zu anderen Komponenten einer Android-App nicht ohne weiteres durch das System zur Schonung der Ressourcen beendet werden kann. Gesteuert wird die Aufnahme über eine grafische Benutzeroberfläche auf dem Smartphone selbst (*AndroidUI*) sowie optional auch über eine Weboberfläche, die über einen einfachen HTTP-Server zur Verfügung steht. In den Benutzeroberflächen können der Name des Probanden sowie die ausgeführte Aktivität angegeben werden. Des Weiteren wird hierüber die Aufnahme gestartet und gestoppt.

Der *CollectionService* delegiert die Aufnahme an den *BandDataRecorder* und den *Local-*

DataRecorder weiter, die jeweils für das Microsoft Band 2 respektive die Smartphone-lokalen Sensoren verantwortlich sind. Über den *BandConnector* wird die Bluetooth-Verbindung zum Band hergestellt und aufrecht erhalten. *BandDataRecorder* und *LocalDataRecorder* zeichnen Daten in Form von *Readings* auf. Die Methode *getValuesPerComponent()* liefert bei einem triaxialen Sensor beispielsweise Daten der Form $\{x \to 1.0, y \to 2.0, z \to 3.0\}$.

Wird die Aufnahme gestoppt, werden die Readings mitsamt den Metadaten Name des Probanden und Aktivität serialisiert und per SFTP auf einen entfernten Speicher hochgeladen. Anschließend werden die Daten von dort abgerufen und wie im folgenden Abschnitt beschrieben weiterverarbeitet.

5.2. Transformation der Daten

Jede einzelne Aufnahme eines Nutzers und einer Aktivität besteht aus einer Reihe von Daten mit Zeitstempeln. In dieser Form kann noch kein konventioneller ML-Klassifikationsalgorithmus mit den Daten arbeiten, da diese Daten in Form einer Liste von *Instanzen* erwarten. Eine Instanz besteht aus einer Menge von *Attributen/Features*, die im Rahmen dieser Arbeit bis auf das nominale *Target*-Attribut allesamt kardinal sind. Das Target-Attribut ist dabei die durchgeführte Aktivität. Entsprechend ist eine Transformation der Daten notwendig, um diese tatsächlich nutzbar zu machen.

Um Instanzen zu erzeugen, teilt die Transformationssoftware die Ursprungsdaten zunächst in Intervalle mit zehnsekündiger Dauer auf. Ein solches Intervall soll schließlich eine Instanz, d.h. ein Beispiel für eine Aktivität bilden. Das erste und das letzte Intervall werden gelöscht, da in dieser Zeit die Aufnahme gestartet und gestoppt wurde, was ansonsten durch den anderen Bewegungsablauf die Ergebnisse verfälschen könnte. Aktuell umfasst das Intervall noch mehrere Readings, die aggregiert werden müssen.

5.2.1. Beschreibung der Aggregatfunktionen

Um aus den sequentiellen Daten des Intervalls eine feste Anzahl von Features zu gewinnen, die dieses Intervall beschreiben, werden Aggregatfunktionen benötigt, die eine Liste von Zahlen $A[1..N]$ auf eine feste Anzahl von Werten reduziert, welche die sequentiellen Daten in Kombination möglichst gut beschreiben. In den folgenden Abschnitten werden die auch von Weiss et al. verwendeten [29] Aggregatfunktionen in Anlehnung an Kwapisz et al. [17] begründet und definiert.

Average & Standard Deviation

Der Durchschnittswert einer Reihe von Zahlen ist im Kontext der Aktivitätenerkennung relevant, da er zusammen mit der ebenfalls nachfolgend definierten Standardabweichung auf einfache Art und Weise angibt, in welchen Bereichen sich die Sensordaten bewegen.

$$\mathrm{Avg}(A) := \frac{1}{N} \cdot \sum_{a \in A} a$$

$$\mathrm{StdDev}(A) := \sqrt{\frac{1}{N} \cdot \sum_{a \in A} (a - \mathrm{Avg}(A))^2}$$

Average Absolute Difference

Diese Funktion gibt ähnlich wie *StdDev* Veränderungen in den Datenreihen wieder, ist dabei jedoch nicht quadratisch und somit für ML-Algorithmen möglicherweise leichter zu handhaben.

$$\text{AvgAbsDiff}(A) := \frac{1}{N} \cdot \sum_{a \in A} |a - \text{Avg}(A)|$$

k-Binned Distribution

Im Gegensatz zu den obigen Funktionen liefert diese Aggregatfunktion einen k-Vektor anstatt eines Skalars. Wie *Avg* und *StdDev* erklärt diese Funktion den Wertebereich der Datenreihen, deckt diesen allerdings vollständig ab. Da die Interpretation der Ergebnisse dieser Funktion dafür komplexer ist, handelt es sich um eine sinnvolle Ergänzung zu den genannten Funktionen.

Seien $i \in \{0, ..., k-1\}, S := \frac{\max A - \min A}{k}$. Dann ist die *Binned Distribution* wie folgt definiert:

$$\text{Bin}(A, i) := \#\{a \in A | (\min A) + i \cdot S \leq a < (\min A) + (i+1) \cdot S\}$$

$\text{Bin}(A, i)$ ist demzufolge die Anzahl der Elemente in Korb i, wenn man A der Größe nach sortiert auf k Körbe verteilt. Der Bin ist hierbei als Multimenge zu verstehen und darf somit Elemente auch mehrfach enthalten.

Average Root of Squares

Auch diese Aggregatfunktion unterscheidet sich von den bisherigen, da sie mehrere Listen als Parameter erhält und somit Zusammenhänge zwischen diesen erklärt:

$$\text{Aros}(A_1, ..., A_M) = \frac{1}{M} \cdot \sum_{i=1}^{M} \sqrt{\sum_{a \in A_i} a^2}$$

Average Time between Peaks

Diese Aggregatfunktion gibt die durchschnittliche Zeit zwischen Höhepunkten in A zurück, die ein signifikantes Merkmal körperlicher Aktivitäten sein können, wofür zusätzlich eine Funktion timeForIndex : $\mathbb{N} \to \mathbb{N}$ benötigt wird. Die Wahl der Höhepunkte erfolgt durch eine Heuristik.

Algorithmus 2 beschreibt die Berechnung des Wertes. Zunächst wird mittels Algorithmus 1 bestimmt, an welchen Indizes in der Liste Werte stehen, die deutlich größer als ihre Nachfolger sind. Dadurch wird definiert, was eine Spitze in dieser spezifischen Liste ausmacht. Anschließend werden Spitzen gesucht, die maximal um den Faktor *threshold* kleiner sind als die größte Spitze, die der Algorithmus gefunden hast. Diese Grenze wird solange gesenkt, bis genügend Spitzen gefunden wurden oder die Grenze so niedrig liegt, dass es sich nicht mehr um Spitzen handelt. Als letztes berechnet der Algorithmus die durchschnittliche Zeit zwischen den Spitzen mit Hilfe der Funktion *timeForIndex*.

Die im Pseudocode verwendeten Konstanten erwiesen sich im praktischen Test an den gesammelten Daten als gut geeignet. Für andere Datensätze kann eine Anpassung erforderlich sein.

Algorithm 1 IndicesOfPeaks(A, t), $t \in [0,1]$. Returns a list of indices i where $A[i] \cdot t \geq A[i+1]$

```
indices ← ∅
for i ∈ {1, ..., |A| − 1} do
    if A[i] · t ≥ A[i + 1] then
        indices ← indices ∪ {i}
    end if
end for
return indices
```

Algorithm 2 AverageTimeBetweenPeaks(A, timeForIndex, minPeaks)

```
▷ First, heuristically define what a peak is
indicesOfPeaks ← IndicesOfPeaks(A, t = 0.8)
if indicesOfPeaks = ∅ then
    return Nil
end if
highestPeakIdx ← Index of highest peak in indicesOfPeaks

▷ Lower the threshold until we have found enough peaks or the threshold is too low
otherPeakIndices ← ∅
threshold ← 0.8
repeat
    otherPeakIndices ← Indices of A where A[i] ≥ A[highestPeakIdx]· threshold
    threshold ← threshold · 0.9
until |otherPeaksIndices| ≥ minPeaks ∨ threshold < 0.3

▷ Check whether we have found enough peaks
if |otherPeakIndices| < minPeaks then
    return Nil
end if

▷ Calculate the average time between the peaks
timesBetweenPeaks ← ∅
for i ∈ {2, ..., |otherPeakIndices|} do
    time ← timeForIndex(otherPeakIndices[i]) - timeForIndex(otherPeakIndices[i − 1])
    timesBetweenPeaks ← timesBetweenPeaks ∪{time}
end for
return AVG(timesBetweenPeaks)
```

5.2.2. Anwendung der Aggregatfunktionen

Die Transformationssoftware reduziert nun mit Hilfe der Aggregatfunktionen die Readings eines jeden Intervalls I auf eine konstante Anzahl skalarer Werte, welche die Features der Instanz darstellen. Im Folgenden sei I ein Array der Struktur $I[G][S][K] \in \mathbb{R}^{\alpha}$, das die Messungen gruppiert nach Gerät (G), Sensor (S) und Komponente (K) beinhaltet. Des Weiteren sei F_I die Menge der Features des Intervalls I. Die Folgenden Unterabschnitte zeigen den Aufbau von F_I mittels der Aggregatfunktionen:

Pro Kombination von Gerät G und Sensor S

Seien P die einzelnen Komponenten des Sensors, beispielsweise $P = \{x, y, z\}$.

$$\begin{aligned} \text{aros} &\leftarrow \text{Aros}(I[G][S][P[1]], ..., I[G][S][P[|P|]]) \\ F_I &\leftarrow F_I \cup \{((G,S), \text{aros})\} \end{aligned}$$

Im Kontext betrachtet liefert *Aros* am Beispiel des Beschleunigungssensors gewissermaßen die durchschnittliche Beschleunigung, die alle Achsen zusammen erfahren haben.

Pro Kombination von Gerät G, Sensor S und Komponente K

Sei *timeForIdx* hier eine Funktion, die den Zeitstempel des jeweiligen Datums im Intervall zurückgibt. Sei außerdem $k := 10$, sodass 10 Körbe verwendet werden, was sich experimentell als geeignet herausstellte.

$$\begin{aligned} F_I &\leftarrow F_I \cup \{((G,S,K), \text{Avg}(I[G][S][K]))\} \\ F_I &\leftarrow F_I \cup \{((G,S,K), \text{StdDev}(I[G][S][K]))\} \\ F_I &\leftarrow F_I \cup \{((G,S,K), \text{AvgAbsDiff}(I[G][S][K]))\} \\ F_I &\leftarrow F_I \cup \{((G,S,K), \text{AverageTimeBetweenPeaks}(I[G][S][K], \text{timeForIdx}, 3))\} \\ F_I &\leftarrow F_I \cup \{((G,S,K), \text{Bin}(I[G][S][K], i))\} \forall i \in \{1, ..., k\} \end{aligned}$$

5.2.3. Beispiel

Wir betrachten für die Aktivität *Jogging* den Beispieldatensatz

$$\begin{aligned} D = \{&(0, \text{Band}, \text{Gyroscope}, x \rightarrow 1.0, y \rightarrow 0.0), \\ &(0, \text{Band}, \text{Accelerometer}, x \rightarrow 1.0, y \rightarrow 1.0), \\ &(1000, \text{Band}, \text{Gyroscope}, x \rightarrow 2.0, y \rightarrow 2.0), \\ &(1000, \text{Band}, \text{Accelerometer}, x \rightarrow 0.0, y \rightarrow 1.0), \\ &(2000, \text{Band}, \text{Gyroscope}, x \rightarrow 3.0, y \rightarrow 0.0), \\ &(2000, \text{Band}, \text{Accelerometer}, x \rightarrow 1.0, y \rightarrow 3.0)\} \end{aligned}$$

Zur Vereinfachung wird von biaxialen Sensoren ausgegangen, sodass keine z-Komponente vorhanden ist. Als Aggregatfunktionen werden der Durchschnitt, die *Binned Distribution* mit

zwei *Bins* und der *Average Root of Squares* verwendet. Die gewählte Intervallgröße beträgt zwei Sekunden, sodass sich die folgenden Intervalle ergeben:

$$\begin{aligned} I_1 = \{&(0, \text{Band}, \text{Gyroscope}, x \to 1.0, y \to 0.0),\\ &(0, \text{Band}, \text{Accelerometer}, x \to 1.0, y \to 1.0),\\ &(1000, \text{Band}, \text{Gyroscope}, x \to 2.0, y \to 2.0),\\ &(1000, \text{Band}, \text{Accelerometer}, x \to 0.0, y \to 1.0)\},\\ I_2 = \{&(2000, \text{Band}, \text{Gyroscope}, x \to 3.0, y \to 0.0),\\ &(2000, \text{Band}, \text{Accelerometer}, x \to 1.0, y \to 3.0)\} \end{aligned}$$

Nun werden die Aggregatfunktionen auf I_1 angewendet:

$$\begin{aligned} &\text{Avg}_{\text{Band, Gyroscope, x}} = (1+2)/2 = 1.5\\ &\text{Avg}_{\text{Band, Gyroscope, y}} = (0+2)/2 = 1\\ &\text{Avg}_{\text{Band, Accelerometer, x}} = (1+0)/2 = 0.5\\ &\text{Avg}_{\text{Band, Accelerometer, y}} = (1+1)/2 = 1\\ &\text{Bin}_{\text{Band, Gyroscope, x}}(0) = \#\{1\} = 1\\ &\text{Bin}_{\text{Band, Gyroscope, x}}(1) = \#\{2\} = 1\\ &\text{Bin}_{\text{Band, Gyroscope, y}}(0) = \#\{0\} = 1\\ &\text{Bin}_{\text{Band, Gyroscope, y}}(1) = \#\{2\} = 1\\ &\text{Bin}_{\text{Band, Accelerometer, x}}(0) = \#\{0\} = 1\\ &\text{Bin}_{\text{Band, Accelerometer, x}}(1) = \#\{1\} = 1\\ &\text{Bin}_{\text{Band, Accelerometer, y}}(0) = \#\{\} = 0\\ &\text{Bin}_{\text{Band, Accelerometer, y}}(1) = \#\{\} = 0\\ &\text{Aros}_{\text{Band, Gyroscope}} = (1/2) \cdot (\underbrace{\sqrt{1^2+2^2}}_{x} + \underbrace{\sqrt{0^2+2^2}}_{y}) \approx 2.12\\ &\text{Aros}_{\text{Band, Accelerometer}} = (1/2) \cdot (\underbrace{\sqrt{1^2+0^2}}_{x} + \underbrace{\sqrt{1^2+1^2}}_{y}) \approx 1.2 \end{aligned}$$

Die Berechnung für I_2 erfolgt analog. Aus den obigen Features ergibt sich der folgende Datensatz, der von einem herkömmmlichen Klassifikationsalgorithmus verarbeitet werden kann:

Intervall	$\text{Avg}_{\text{Band, Gyroscope, x}}$	$\text{Avg}_{\text{Band, Gyroscope, y}}$	...	sampleClass
I_1	1.5	1	...	Jogging
I_2	...	...	...	Jogging

Die Intervallspalte dient nur der Illustration und wird dem ML-Algorithmus nicht gezeigt. Die grundlegende Transformation in ein zu konventionellen ML-Algorithmen kompatibles Format ist damit abgeschlossen. Der folgende Abschnitt behandelt die Nachbearbeitung dieser Daten, welche die Verarbeitung durch ML-Algorithmen erleichtert.

5.3. Nachbearbeitung der Daten

Bei der Transformation kann sich das Problem ergeben, dass nicht jede Zelle der resultierenden Tabelle einen Wert enthält. Dazu kann es beispielsweise kommen, wenn der Algorithmus `AverageTimeBetweenPeaks` *Nil* zurückgibt. Enthält eine Zelle keinen Wert, so wird in ihr der Durchschnittswert des gefüllten Teils der Spalte eingesetzt. Um dem Risiko vorzubeugen, dass ML-Algorithmen Spalten mit betragsmäßig hohen Werten fälschlicherweise auch automatisch hohen Einfluss zuschreiben, werden alle numerischen Spalten mit Hilfe des z-Score-Verfahrens normalisiert. Dazu wird von den Werten jeder Spalte der Durchschnittswert dieser abgezogen, wonach das Resultat durch die Standardabweichung der Spalte geteilt wird. Der Betrag einer Zelle gibt danach an, wie viele Standardabweichungen vom Durchschnitt abgewichen wird. Nun werden mit den Daten wie im folgenden Abschnitt beschrieben mithilfe von ML-Algorithmen Modelle trainiert.

5.4. Anwendung von ML-Klassifikationsalgorithmen

Abbildung 5.2 zeigt noch einmal abschließend den Aufnahme- und Transformationsprozess bis zu diesem Schritt. Erst in der Tabellenform, die im untersten Teil der Abbildung zu sehen ist, können die gesammelten Daten von einem herkömmlichen ML-Klassifikationsalgorithmus zum Training und außerdem zur Evaluation des damit gelernten Modells verwendet werden.

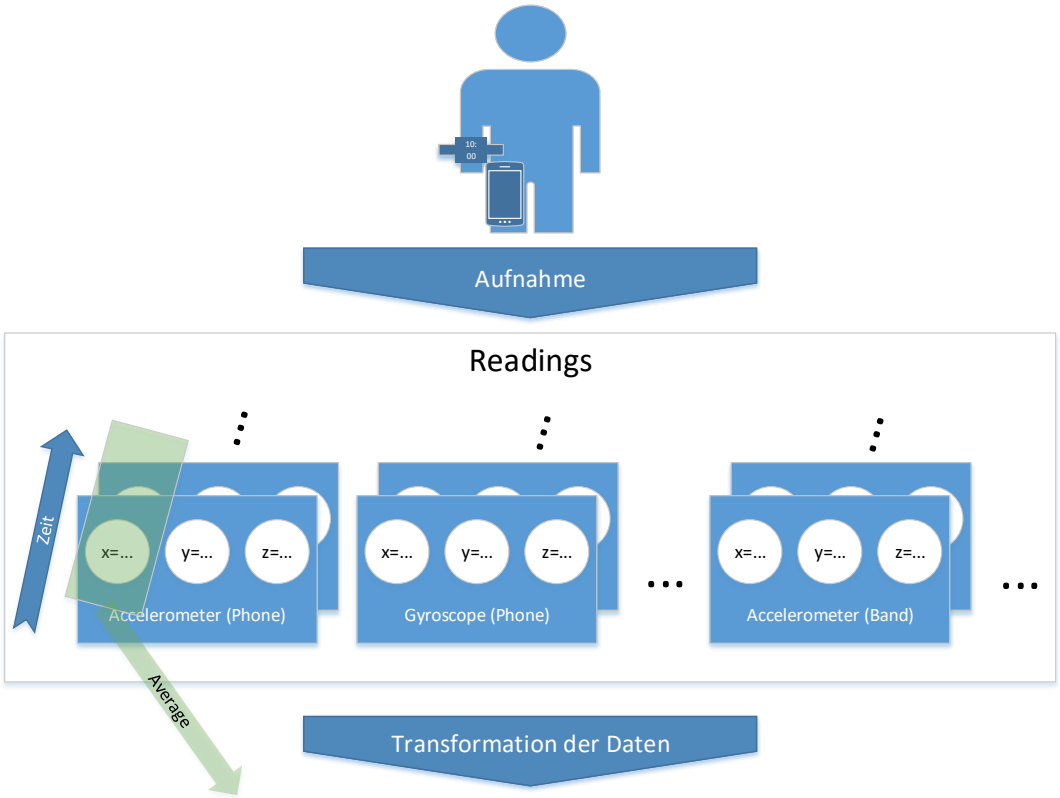
Um die Vergleichbarkeit mit [29] sicherzustellen, wurde die Java-basierte ML-Software WEKA zum maschinellen Lernen verwendet. Diese bietet fertige Implementierungen diverser ML-Algorithmen an, die sowohl mittels einer grafischen Benutzeroberfläche, als auch in Code verwendet werden können. Die in dieser Arbeit verwendeten Algorithmen entsprechen denen aus [29] und werden im Folgenden kurz beschrieben, bevor die Evaluierung der daraus resultierenden Modelle erfolgt. Falls nicht anders angegeben, wurden für jedes Modell die standardmäßigen Hyperparameter verwendet. Dabei handelt es sich um Parameter, die der jeweilige Algorithmus nicht selbst lernen kann.

5.4.1. Random Forest (RF)

Ein *Random Forest* ist ein Wald von Entscheidungsbäumen (erklärt in Kapitel 14.4 [4]), die randomisiert entstanden sind. Soll eine neue Instanz klassifiziert werden, treffen alle Entscheidungsbäume eine Entscheidung und die am häufigsten vorhergesagte Klasse gewinnt. WEKA nutzt eine Implementierung nach [5]. Experimentell haben sich die folgenden Hyperparameterwerte als sinnvoll erwiesen: Die Anzahl der Bäume in Wald wurde auf 50, die Anzahl der verwendeten Features pro Baum auf 10 sowie die maximale Tiefe auf 25 festgelegt.

5.4.2. J48-Entscheidungsbaum (J48)

J48 ist eine Implementierung des *C4.5*-Algorithmus [24]. Der Entscheidungsbaum wird anhand von Trainingsdaten T iterativ durch das Hinzufügen von Entscheidungsknoten erzeugt, die den Informationsgehalt der Teilmengen von T minimieren, die aus dem Split an jenem Entscheidungsknoten hervorgehen. Der Informationsgehalt ist umso höher, je mehr verschiedene Klassen sich in einer Menge von Instanzen befinden. Ein niedriger Informationsgehalt besagt



Intervall	Avg_ACC_Phone_X	...	StdDev_ACC_Phone_X	...	Avg_ACC_Band_X	...	Aktivität
[t,t+10s)	0,5	...	-0,25	...	0,001	...	Jogging
[t+10s,t+20s)	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...

Abbildung 5.2.: Aufnahme- und Transformationsprozess

demnach, dass nach einer Entscheidung an einem Entscheidungsknoten klarer geworden ist, welcher Klasse die Instanzen in den Mengen nach dem Split zuzuordnen sind.

5.4.3. Instance-Based-Learner (IB*k*)

Der IBk-Algorithmus ist ein kNN-basierter Klassifikationsalgorithmus. Soll einer neuen Instanz $x \in \mathbb{R}^n$ eine Klasse zugeordnet werden, werden anhand eines Distanzmaßes für den Vektorraum $\mathbb{R}^n$ die k nächsten Nachbarn von x gesucht. x wird danach die Klasse zugeordnet, die unter den k Nachbarn am häufigsten vorkam. Weitere Details sind zu finden in [1]. k ist dabei ein Hyperparameter, der auf 3 eingestellt wurde.

5.4.4. Naive Bayes (NB)

Naive Bayes ist eine probabilistische Methode zur Klassifikation [14]. Mit $x = (x_1, ..., x_n)$ wird die Klasse C gesucht, die $\mathbb{P}(C|x_1, ..., x_n)$ maximiert. x besteht hier aus nominalen Attributen. Der Satz von Bayes besagt, dass $\mathbb{P}(C|x) = \frac{\mathbb{P}(C)\mathbb{P}(x|C)}{\mathbb{P}(x)}$. Da über C optimiert wird, kann der

Nenner für diesen Ansatz ignoriert werden. Nimmt man *naiv* an, dass $x_1, ..., x_n$ voneinander unabhängig sind, kann somit statt $\mathbb{P}(C|x)$ auch folgender Term über C maximiert werden: $\mathbb{P}(C)\prod_{i=1}^{n}\mathbb{P}(x_i|C)$. Durch den Trainingsdatensatz sind die $\mathbb{P}(C)$ als Prior der Klassen C sowie alle $\mathbb{P}(x_i|C)$ bekannt, sodass der Term durch einfaches Ausprobieren aller Klassen maximiert werden kann.

Enthält x numerische Attribute, müssen diese zunächst durch nominale Attribute ersetzt werden. Der Wertebereich aller x_i über den Trainingsdatensatz hinweg kann in eine feste Anzahl von Intervallen aufgeteilt werden. Für jedes Intervall I wird nun eine Indikatorvariable als Feature angelegt, die angibt, ob für eine Instanz x gilt, dass $x_i \in I$.

5.4.5. Multilayer Perceptron (MLP)

Bei dieser Methode handelt es sich um ein mehrschichtiges Netzwerk von sogenannten *Perzeptren*, auch *(künstliches) neuronales Netzwerk* genannt. Ein Perzeptron erhält n Eingaben $x_1, ..., x_n$ und gewichtet diese mit Parametern $\Theta_1, ..., \Theta_n$, die gelernt werden. Das Perzeptron berechnet $y = g(\sum_{i=1}^{n}\Theta_i x_i)$ als Ausgabewert, wobei g eine Aktivierungsfunktion ist, die den Ausgabewert beschränkt, beispielsweise die Sigmoidfunktion.

Die Ausgabe eines Perzeptrons kann einem Perzeptron der nächsten Schicht als Eingang dienen. Die letzte Schicht fungiert als Ausgabe des Netzwerks, dessen Fehler anhand eines Trainingsdatensatzes bestimmt werden kann. Jedes Perzeptron berechnet, wie die eigenen Parameter Θ angepasst werden müssen, um den Fehler zu minimieren, wofür der Fehler der Perzeptren der letzten Schicht an die vorherige Schicht weiter propagiert wird (*Backpropagation*). Zur Optimierung der Parameter wird ein stochastischer Gradientenabstieg verwendet. Zur Klassifikation können die Ausgangswerte der Perzeptren der letzten Schicht als Indikatorvariablen für die vorhandenen Klassen interpretiert werden. Weitere Informationen sind zu finden in [12].

6. Evaluation

Dieses Kapitel widmet sich der Auswertung der Ergebnisse anhand verschiedener Metriken. Eingebettet in der Transformationssoftware befindet sich ein Präprozessor, der die aufgenommenen Daten vor der Transformation manipulieren kann. Auf diese Weise werden mehrere Trainingsdatensätze erstellt, die in der Evaluation daraufhin analysiert werden, wie genau ein Klassifikator nach dem Training mit ihnen Vorhersagen treffen kann.

Die folgenden Datensätze werden generiert:

1. *Combined*: Alle Daten werden ohne Veränderungen mit in die Transformation einbezogen.

2. *NoisyBandTimestamps*: Die Zeitstempel der Readings des Band werden mit gaußschem Rauschen versehen.

3. *Band combined*: Alle Daten des Smartphones werden vor der Transformation verworfen, sodass nur noch die Daten des Band verbleiben..

4. *Band accel*: Nur die Daten des Beschleunigungssensors des Microsoft Band 2 werden mit in die Transformation einbezogen.

5. *Band gyro*: Nur die Daten des Gyroskops des Microsoft Band 2 werden mit in die Transformation einbezogen.

6. *Phone comb*: Alle Daten des Microsoft Band 2 werden vor der Transformation verworfen, sodass nur noch die Daten des Smartphones verbleiben.

7. *Phone accel*: Nur die Daten des Beschleunigungssensors des Smartphones werden mit in die Transformation einbezogen.

8. *Phone gyro*: Nur die Daten des Gyroskops des Smartphones werden mit in die Transformation einbezogen.

9. *SamplingRate1Hz*: Es werden Readings verworfen, als hätten Band und Smartphone jeweils nur Readings bei einer Rate von 1 Hz geliefert.

10. *SamplingRate5Hz*: Es werden Readings verworfen, als hätten Band und Smartphone jeweils nur Readings bei einer Rate von 5 Hz geliefert.

Algo.	Phone accel	Phone gyro	Band accel	Band gyro	Comb	Band comb	Phone comb
RF	88.7	**68.2**	**91.6**	**80.5**	**99.4**	**97.1**	**90.7**
J48	**90.0**	64.3	86.5	72.7	92.8	90.1	89.7
IB3	62.2	44.8	77.0	59.4	82.3	76.9	60.1
NB	87.6	60.4	90.9	78.8	96.2	92.2	85.5
MLP	78.9	51.5	88.9	67.8	94.8	90.7	75.1
∅	81.5	57.8	87.0	71.8	93.1	89.4	80.2

Tabelle 6.1.: Genauigkeit der persönlichen Modelle in Prozent

6.1. Effektivität der Datenkombination

Dieser Abschnitt widmet sich der Frage, wie effektiv die Kombination der Daten von Band und Smartphone hinsichtlich der Vorhersagegenauigkeit gegenüber einer einzigen Datenquelle ist. Weiss et al. werten in ihrem Papier die Genauigkeit von Modellen aus, die entweder auf Daten des Beschleunigungssensors eines Smartphones, des Beschleunigungssensors einer Smartwatch oder des Gyroskops einer Smartwatch zurückgreifen [29]. Eine Kombination der Daten schlagen die Autoren vor, führen diese allerdings nicht durch.

Tabelle 6.1 zeigt die Genauigkeit der persönlichen Modelle und Tabelle 6.2 die der unpersönlichen Modelle. Ein persönliches Modell basiert auf Daten des jeweiligen Benutzers, während ein unpersönliches Modell noch keine Daten des Nutzers gesehen hat, für den eine Vorhersage getroffen werden soll. Die *Genauigkeit* ist definiert als der Anteil der korrekt klassifizierten Instanzen, die dem Modell zum Test vorgelegt wurden.

Die erste Spalte der in diesem Abschnitt referenzierten Tabellen gibt den Algorithmus an, der zur Klassifikation verwendet wurde. Eine Erläuterung der Algorithmen ist in Abschnitt 5.4 zu finden. Fettgedruckt ist immer der jeweils höchste Wert einer Spalte.

6.1.1. Persönliche Modelle

Die Evaluierung der persönlichen Modelle wird wie folgt durchgeführt: Für jeden Benutzer b wird eine 10-fache Kreuzvalidierung durchgeführt, wobei die Trainingsdaten nur Daten von b beinhalten. Anschließend wird der Durchschnitt der einzelnen Genauigkeiten über alle Benutzer gebildet, um einen globalen Genauigkeitswert zu erhalten. Um dieses Testverfahren zu ermöglichen, wurde die WEKA-Software leicht angepasst: Die Methode zur Kreuzvalidierung wurde um einen Parameter `foldSelector` : `Instanzen` $\rightarrow \mathbb{N}$ erweitert, der für eine Instanz aus dem Datensatz zurückgibt, in welcher Iteration der Kreuzvalidierung diese aus dem Trainingsdatensatz ausgeschlossen und stattdessen im Testdatensatz verwendet wird. Die Ergebnisse dieser Evaluierung befinden sich in Tabelle 6.1.

Nutzt man nur je ein Gyroskop, ist die Qualität der persönlichen Modelle am schlechtesten. Im Gegensatz dazu haben bereits Modelle, die lediglich Daten von einem der Beschleunigungssensoren verwenden, eine gute Aussagekraft.

Vergleicht man *Band accel* als die beste einzelne Datenquelle mit den Kombinationsmöglichkeiten, so lässt sich feststellen, dass die durch die Hinzunahme der anderen Sensoren des Band resultierende Kombination eine Genauigkeit von 97.1% erzielt, was einer Verbesserung um 5.5 Prozentpunkte entspricht. Im Gegensatz dazu liefert die Kombination der beiden Smartphone-Sensoren lediglich eine Verbesserung von 0.7 Prozentpunkte gegenüber *Phone*

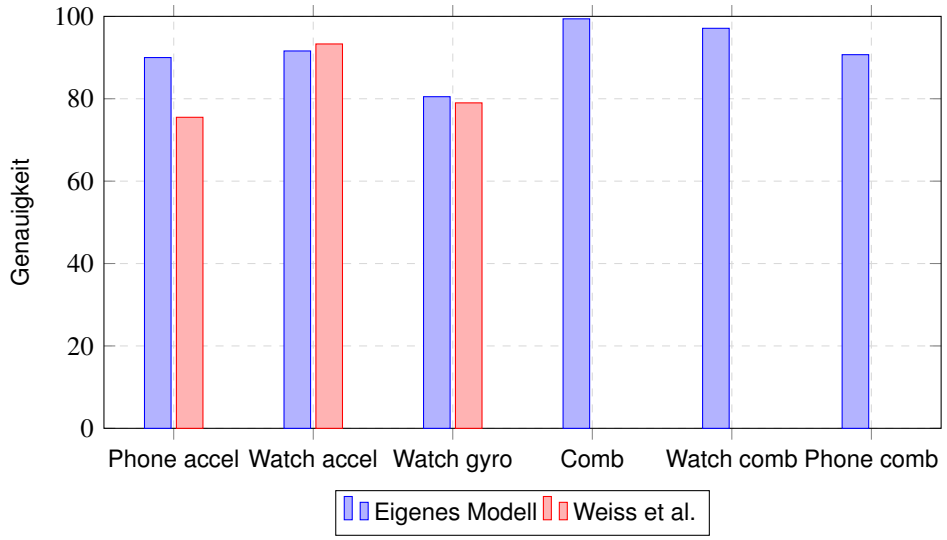


Abbildung 6.1.: Vergleich der Genauigkeiten der persönlichen Modelle mit Weiss et al. [29]. Die *Watch* ist in dieser Arbeit das Band.

accel. Der einzelnen Quelle *Band accel* ist selbst die Kombination der Smartphone-Sensoren unterlegen.

Hebt man jedoch diese künstlichen Einschränkungen auf, so wird eine Genauigkeit von 99.4% erreicht, sodass Benutzer nach dreiminütiger Aufnahme einer Aktivität davon ausgehen könnten, dass diese in Zukunft bis auf wenige Ausreißer automatisch erkannt wird. Insgesamt konnte gegenüber *Phone Accel* damit eine Verbesserung um 9.4 Prozentpunkte erreicht werden. Gegenüber dem besten Ergebnis eines einzelnen Sensors (*Band accel*) mit 91.6% liefert die Kombination aller Daten eine Verbesserung um 7.8 Prozentpunkte.

Anzumerken ist, dass die Genauigkeit von 97.1%, die durch die Kombination der Band-Sensoren erzielt wird, je nach Anwendungszweck bereits ausreichend sein kann und die Hinzunahme von Smartphone-Sensoren daher nicht unbedingt erforderlich ist. Dennoch folgt aus diesen Ergebnissen die wertvolle Erkenntnis, dass zumindest die Kombination der verschiedenen Sensoren des Band eine nennenswerte Verbesserung der Genauigkeit erzielt.

Vergleich mit Weiss et al.

Da diese Bachelorarbeit auf der Arbeit von Weiss et al. [29] aufbaut, ist ein Vergleich der Ergebnisse sinnvoll. Abbildung 6.1 zeigt die Genauigkeiten der besten Modelle in Abhängigkeit von den Datenquellen. Sowohl die Quellen *Watch accel* als auch *Watch gyro* liefern sehr ähnliche Ergebnisse wie bei Weiss et al., nur das *Phone accel*-Modell in dieser Arbeit liefert im direkten Vergleich eine deutlich bessere Genauigkeit. Eine mögliche Erklärung wäre die im Vergleich mit Weiss et al. 10-fach höhere Sampling-Rate des Sensoren des Smartphones gewesen, jedoch lieferte ein Test mit einem RF-Modell und auf 20 Hz beschränkten Smartphone-Daten immerhin noch eine Genauigkeit von 88.2%. Da dies dennoch 12.7 Prozentpunkte über dem *Phone Accel*-Modell von Weiss et al. liegt, lässt sich diese Erklärung ausschließen. Eine weitere Erklärungsmöglichkeit wäre, dass die Messunsicherheit des Sensors im OnePlus 3 unter der

Algo.	Phone accel	Phone gyro	Band accel	Band gyro	Comb	Band comb	Phone comb
RF	**39.2**	**35.1**	**75.2**	**61.7**	**78.5**	**76.5**	**41.3**
J48	32.9	29.1	61.6	49.2	59.7	61.3	32.6
IB3	24.3	22.6	60.9	45.0	52.9	58.0	26.0
NB	31.8	30.0	67.3	56.9	60.9	62.3	35.0
MLP	28.8	29.3	70.3	53.5	54.9	74.3	31.5
∅	81.5	29.2	67.1	53.3	61.4	66.5	33.3

Tabelle 6.2.: Genauigkeit der unpersönlichen Modelle in Prozent

des Sensors im Samsung Galaxy S4 liegt, das von Weiss et al. zur Aufnahme genutzt wird, jedoch kann diese Hypothese mangels eines solchen Gerätes nicht verifiziert werden. Die Methodik zur Reduktion der Sampling-Rate wird in Abschnitt 6.3 erklärt, der sich weiteren solchen Analysen widmet.

Erwartungsgemäß ist die Genauigkeit mit den vollständig kombinierten Datenquellen arbeitsübergreifend am höchsten, jedoch ist dies sogar der Fall, wenn man die Daten des Smartphones entfernt. Nutzt man hingegen nur die Daten des Smartphones, ist die Genauigkeit vergleichbar mit den beiden *Watch accel*-Ergebnissen.

Variation zwischen Teilnehmern

Motiviert durch Weiss et al. (2012) [30] wurde auch die Genauigkeitsverteilung der Modelle untersucht. Es ist von Interesse, wie die Genauigkeit unter den Teilnehmern variiert, da allein eine hohe Durchschnittsgenauigkeit nicht zeigt, wie praxistauglich die Modelle tatsächlich sind. Ist die Genauigkeit für einige Teilnehmer sehr schlecht, für andere Teilnehmer hingegen sehr gut und damit insgesamt instabil, schränkt dies die Verwendbarkeit der Methode ein, da die Zuverlässigkeit gering ist.

Eine Analyse der persönlichen *Comb*-Modelle ergab jedoch, dass keines eine schlechtere Genauigkeit als 98.69% lieferte, wobei bereits der außerordentlich hohe Durchschnittswert von 99.4% implizierte, dass die persönlichen Modelle nicht schlecht sein konnten. Eine größere Rolle wird diese Analyse demnach bei den unpersönlichen Modellen spielen.

6.1.2. Unpersönliche Modelle

Für die Evaluierung der unpersönlichen Modelle wird eine Kreuzvalidierung über die Nutzer durchgeführt. Das heißt der Datensatz D wird für alle Benutzer b aufgeteilt in $D = \text{Train} \uplus \text{Test}$ mit $\text{Train} = \{\text{Intervall ist nicht von } b\}, \text{Test} = D \backslash \text{Train}$. Tabelle 6.2 ist das Ergebnis dieser Evaluierung.

Das schlechteste Ergebnis wird mit 35.1% bei den unpersönlichen Modellen erzielt, wenn nur das Gyroskop des Smartphones verwendet wird. Nur wenig besser sind die Modelle, die nur den Beschleunigungssensor des Smartphones (39.2%) oder beide Sensoren des Smartphones (41.3%) nutzen. Für den praktischen Gebrauch sind diese Modelle aufgrund ihrer Ungenauigkeit untauglich. Schon rund 20 Prozentpunkte besser sind die Modelle, die nur das Gyroskop des Band nutzen, wobei der Wechsel zum Beschleunigungssensor eine weitere Verbesserung um fast 14 Prozentpunkte ermöglicht. Offenbar erklären die Daten des Beschleunigungssensors einen Großteil der Daten, die auch durch das Gyroskop und die anderen

Sensoren des Band erklärt werden, sodass eine Kombination aller Band-Sensoren lediglich ein Verbesserung um rund einen Prozentpunkt ermöglicht. Selbiges gilt auch für die Hinzunahme der Daten des Smartphones: Es wird eine Verbesserung um 2 Prozentpunkte erzielt, was darauf schließen lässt, dass die Armbewegungen unter den Probanden ähnlich sind, die Bewegungen auf Hüfthöhe hingegen weniger. Gegenüber dem besten Ergebnis eines einzelnen Sensors (*Band accel*) mit 75.2% liefert die Kombination aller Daten eine Verbesserung um 3.3 Prozentpunkte.

Ähnlich wie bei den persönlichen Modellen zeigt sich auch hier, dass die Hinzunahme von Smartphone-Daten zu allen Band-Daten zumindest hinsichtlich der Durchschnittsgenauigkeit keine signifikante Verbesserung erzielt. Im Gegensatz zu den persönlichen Modellen sorgt allerdings auch die Kombination aller Sensoren des Bands gegenüber *Band accel* nicht für nennenswert höhere Genauigkeiten.

Als denkbare Erklärung für die Schwäche der unpersönlichen Modelle gegenüber den persönlichen Modellen wurde die Hypothese evaluiert, dass persönliche Modelle eher über die Bewegungsrichtung und unpersönliche Modelle eher über die Varianz in den Daten Vorhersagen bilden. Diese Hypothese beruht auf der Tatsache, dass die Ausrichtungen des Smartphones und des Fitness-Trackers über die Teilnehmer hinweg nicht gleich war und beispielsweise eine Vorwärtsbewegung daher sowohl mit einem positiven als auch einem negativen Ausschlag in den Daten des Beschleunigungssensors in den Aufnahmen erscheinen kann. Um dies zu überprüfen wurde durch Manipulation der Daten simuliert, dass Teilnehmer des Experiments in mehreren Durchläufen die Ausrichtung der Geräte ändern. Dazu wurde für jedes aufgenommene Intervall pro Komponente der Beschleunigungssensoren und der Gyroskope der beiden Geräte zufällig entschieden, ob dessen Sensordaten in diesem Intervall mit (-1) oder mit 1 multipliziert werden. Nach dieser Manipulation erreichten unpersönliche RF-Modelle eine Genauigkeit von 76.2% und persönliche RF-Modelle 97.1%, was jeweils einer Beeinträchtigung um 2.3 Prozentpunkte entspricht. Dies bedeutet einerseits, dass das in dieser Bachelorarbeit vorgestellte Verfahren gegenüber der Neuausrichtung der Geräte resistent ist. Andererseits deutet die geringe Schwächung der Modelle auch darauf hin, dass die Hypothese nicht zutreffend ist.

Vergleich mit Weiss et al.

Wie in Abbildung 6.2 ersichtlich wird, unterscheiden sich die unpersönlichen Modelle im Gegensatz zu den persönlichen Modellen allesamt um etwa 5 Prozent von denen von Weiss et al.

Wie bei den persönlichen Modellen ist auch hier die Genauigkeit mit den vollständig kombinierten Daten sowie den kombinierten Daten der Smartwatch, beziehungsweise des Fitness-Trackers arbeitsübergreifend am höchsten.

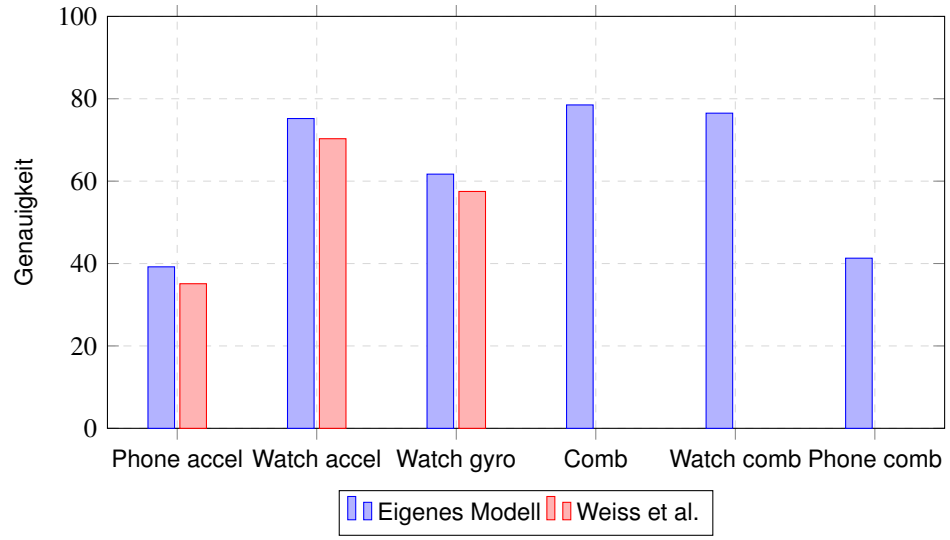


Abbildung 6.2.: Vergleich der Genauigkeiten der unpersönlichen Modelle mit Weiss et al. [29]. Die *Watch* ist in dieser Arbeit das Band.

Variation zwischen Teilnehmern

Wie die persönlichen Modelle wurden auch die unpersönlichen Modelle zusätzlich pro Teilnehmer evaluiert, um eine Genauigkeitsverteilung zu erhalten.

Abbildung 6.3 zeigt die Genauigkeitsverteilungen, die aus dem Experiment hervorgingen. Kein unpersönliches *Comb*-Modell für einen Teilnehmer war ungenauer als 65%, während 80% der Teilnehmer sogar *Comb*-Modelle mit einer Genauigkeit von mindestens 75% besaßen. Es ist auch in dieser Abbildung offensichtlich, dass die *Band accel*-Modelle genauer als die *Phone accel*-Modelle sind, jedoch liegt die Spannweite der Genauigkeiten relativ gleichmäßig zwischen 60 und 90 Prozent, während bei den *Comb*-Modellen eine klare Häufung im Intervall $[75,80)$ zu erkennen ist. Des Weiteren ist im Gegensatz zu den *Comb*-Modellen ein *Band accel*-Modell schlechter als 65% und keines besser als 90%.

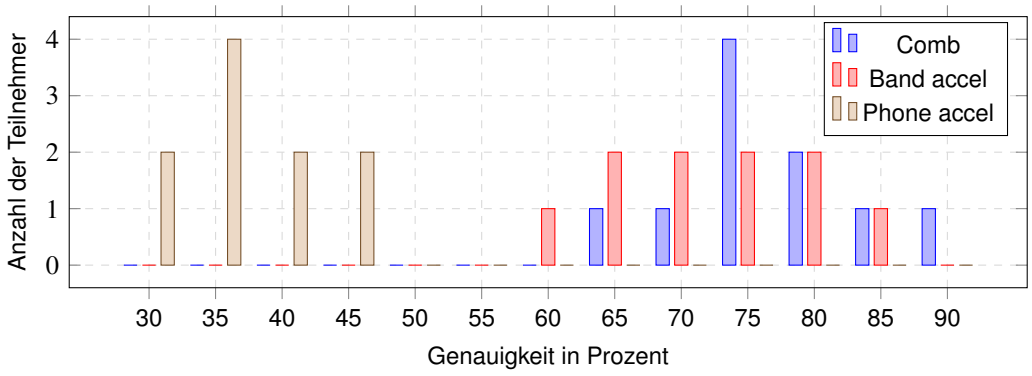


Abbildung 6.3.: Genauigkeitsverteilungen unpersönlicher Modelle. Die Beschriftungen n der x-Achse stehen für die Intervalle $[n,n+5)$.

6.1.3. Anmerkung zu hybriden Modellen

Weiss und Lockhart untersuchten 2012 den Einfluss der Modell-Personalisierung auf die Genauigkeit [30]. Neben persönlichen und unpersönlichen Modellen analysierten sie auch sogenannte *hybride* Modelle, die für einen Benutzer b sowohl auf Daten von b, als auch auf Daten anderer Benutzer zurückgreifen. Dabei stellten sie jedoch fest, dass die Qualität der hybriden Modelle unter der Qualität der persönlichen Modelle lag, weshalb sie keinen Nutzen in diesen Modellen sahen. Ausgenommen haben sie dabei lediglich den Fall, in dem zu einem Benutzer nur sehr wenige Daten bekannt sind. Aufgrund der schlechten Erfolgsaussichten zur Verbesserung der Genauigkeit durch solche Modelle wurde in dieser Arbeit auf eine Untersuchung dieser verzichtet.

6.2. Genauigkeit bei ungenauen Zeitstempeln

Algo.	Comb, Pers.	Verrauscht, Pers.	Comb, Unpers.	Verrauscht, Unpers.
RF	**99.4**	**99.2**	**78.5**	**78.3**
J48	92.8	93.5	59.7	57.3
IB3	82.3	81.5	52.9	52.9
NB	96.2	95.7	60.9	60.5
MLP	94.8	94.2	54.9	55.4
∅	93.1	92.8	61.4	60.9

Tabelle 6.3.: Genauigkeit der Modelle mit verrauschten Band-Zeitstempeln in Prozent

Die Problemstellung, Daten mehrerer Quellen miteinander zu kombinieren, wirft die Frage auf, inwiefern die Synchronisierung der Daten einen Einfluss auf die Genauigkeit der resultierenden Modelle besitzt. Smartphone und Band besitzen je eine eigene Uhr, sodass die Datenquellen zum selben Zeitpunkt t *Readings* mit Zeitstempeln $t + \varepsilon_{\text{Smartphone}}$, bzw. $t + \varepsilon_{\text{Band}}$ mit $|\varepsilon_{\text{Smartphone}} - \varepsilon_{\text{Band}}| > 0$ liefern könnten. Wird die Differenz zu groß, schadet die Kombination der Quellen möglicherweise mehr als sie nützt.

Ein erster Ansatz, um dieses Problem zu bekämpfen, war daher zunächst, die beiden Uhren via Bluetooth und einem Verfahren wie dem *Network Time Protocol (NTP)* [20] zu synchronisieren. Leider ermöglicht das Microsoft Band SDK nicht die Ausführung beliebigen Codes auf dem Band, sondern nur das Abonnieren von Sensordaten-Events, weshalb kein direkter Zugriff auf die Uhr des Gerätes möglich ist. Die Events sind zwar mit Zeitstempeln versehen, jedoch kann die Übertragungszeit nicht ermittelt werden. Eine manuelle Synchronisierung der Uhren ist daher nicht möglich.

Um zu prüfen, inwiefern die Modelle durch eine mögliche, nicht detektierbare Differenz der Uhren beeinträchtigt werden, generiert die Transformationssoftware zusätzlich zum normalen vollständigen Datensatz auch einen, in dem die Zeitstempel des Bands additiv mit gausschem Rauschen versehen wurden. Die Parameter des gausschen Rauschens wurden auf einen Mittelwert von 0 und eine Standardabweichung von 500ms festgelegt. Die Ergebnisse der Auswertung werden in Tabelle 6.3 aufgeführt. Betrachtet man die jeweils besten Modelle, so lässt sich sowohl bei den persönlichen, als auch bei den unpersönlichen Modellen eine

Verschlechterung um lediglich 0.2 Prozentpunkte feststellen. Auch die anderen Modelle sind im Durchschnitt mit Verschlechterungen um 0.3, bzw. 0.5 Prozentpunkte hinreichend resistent gegen das Rauschen.

Um zu prüfen, ob eine Standardabweichung von 500ms sinnvoll ist, wurde über drei Tage hinweg um jeweils 10, 15 und 18 Uhr geprüft, mit welchem zeitlichen Abstand das Umspringen von der vollen Stunde auf die nächste Minute erfolgte. Eine Verzögerung war in keinem der Fälle mit dem Auge beobachtbar, weshalb davon auszugehen ist, dass die Uhren aufgrund der bereits bestehenden Synchronisierung durch die vom Hersteller gelieferte Band-App für Android in der Regel weniger als 500ms voneinander abweichen. Dies führt zu der Annahme, dass die Genauigkeit der Zeitstempel im Rahmen dieser Arbeit keinen negativen Einfluss auf die Ergebnisse hat und somit nicht weiter betrachtet werden muss.

6.3. Genauigkeit bei einer niedrigen Sampling-Rate

Da beide verwendeten Geräte primär akkubetrieben sind, muss bei einer marktreifen Umsetzung einer Aufnahmesoftware auch bedacht werden, dass das Sammeln der Daten Energieressourcen beansprucht. Eine Rolle beim Energieverbrauch spielt auch die Sampling-Rate (Abtastrate): Ist sie hoch eingestellt, können die entsprechenden Sensoren nicht in einen Energiesparmodus wechseln und außerdem müssen von der anfordernden Anwendung mehr Daten verarbeitet werden, wodurch wiederum Anforderungen an den Prozessor gestellt werden [15]. Beim Band kommt hinzu, dass eine höhere Bluetooth-Datenrate in Kauf genommen werden muss, die insbesondere auf den bauartbedingt kleinen Akku des Fitness-Trackers einen negativen Einfluss hat. Aus diesem Grund muss untersucht werden, inwiefern die Sampling-Rate einen Einfluss auf die Genauigkeit der Modelle besitzt, um möglicherweise Energie sparen zu können.

Gerät	Sensor	Durchsch. Sampling-Rate
Smartphone	Accelerometer	200 Hz
Smartphone	Gyroskop	200 Hz
Fitness-Tracker	Accelerometer	8 Hz
Fitness-Tracker	Gyroskop	8 Hz
Fitness-Tracker	Laufgeschwindigkeit	1 Hz
Fitness-Tracker	Hautwiderstand	5 Hz

Tabelle 6.4.: Sampling-Raten der Sensoren

Die Sampling-Raten, mit denen die Aufnahmesoftware Daten aufgezeichnet hat, befinden sich in Tabelle 6.4. Um Sampling-Raten miteinander zu vergleichen, wurden dieselben Rohdaten verwendet wie im normalen, vollständigen Modell. Um eine Sampling-Rate von k Hz zu simulieren, wurden nach einem behaltenen Reading $(t_0\,\text{secs}, \text{sensor}, \text{source}, ...)$ alle weiteren Readings $(t\,\text{secs}, \text{sensor}, \text{source}, ...)$ mit $t \in [t_0, t_0 + 1/k]$ gelöscht. Abbildung 6.4 zeigt beispielhaft Aliasingeffekte, die durch dieses Verfahren entstehen können, wenn Sensordaten Schwingungen beinhalten. Reduziert man die Sampling-Rate, ohne darüberliegende Frequenzen aus dem Signal herauszufiltern, können die noch vorhandenen Daten auf mehrere Arten

(miss-)interpretiert werden. Aus diesem Grund geben die folgenden Ergebnisse untere Schranken für die Genauigkeiten an, die mit dem vorhandenen Datensatz bei verringerten Abtastraten möglich sind.

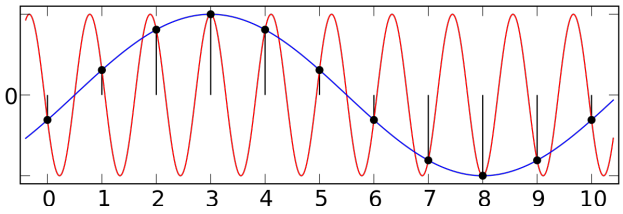


Abbildung 6.4.: Aliasingeffekte, aus Wikipedia [22]

Nun werden die besten Modelle mit Hilfe von Tabelle 6.5 miteinander verglichen. Zur Erinnerung: *Comb, Pers.* erzielte eine Genauigkeit von 99.4% und *Comb, Unpers.* eine Genauigkeit von 78.5%. Reduziert man die Sampling-Rate des persönlichen Modells auf 5 Hz, so reduziert sich die Genauigkeit um lediglich 0.4 Prozentpunkte. Die Reduzierung auf 1 Hz verursacht eine Verschlechterung um 1.4 Prozentpunkte, sodass selbst bei niedrigem Energieverbrauch mit persönlichen Modellen noch sehr gute Ergebnisse erzielt werden können. Stärker beeinträchtigt werden durch die Reduzierung der Sampling-Rate die unpersönlichen Modelle. Beschränkt man diese auf 5 Hz, so ergibt sich gegenüber *Comb, Unpers.* eine Verschlechterung um 3.9 Prozentpunkte. Bei einer Beschränkung auf 1 Hz beträgt die Verschlechterung 8.5 Prozentpunkte, sodass eine Reduzierung der Sampling-Rate auf 1 Hz für unpersönliche Modelle unratsam ist, während mit 5 Hz noch akzeptable Genauigkeiten erzielt werden können.

Algo.	**5 Hz, Pers.**	**1 Hz, Pers.**	**5 Hz, Unpers.**	**1 Hz, Unpers.**
RF	**99.0**	**98.0**	**74.6**	**70.0**
J48	93.2	92.6	62.8	53.4
IB3	77.9	62.0	50.6	37.3
NB	94.6	80.3	63.5	55.0
MLP	92.3	84.4	65.9	46.3
∅	91.4	83.5	63.5	52.4

Tabelle 6.5.: Genauigkeit der Modelle mit reduzierter Sampling-Rate in Prozent

Algo.	Comb, Pers.	Überlapp., Pers.	Comb, Unpers.	Überlapp., Unpers.
RF	**99.4**	**99.2**	**78.5**	**73.4**
J48	92.8	96.3	59.7	57.5
IB3	82.3	81.8	52.9	50.7
NB	96.2	95.8	60.9	59.7
MLP	94.8	95.0	54.9	59.4
∅	93.1	93.6	61.4	60.9

Tabelle 6.6.: Genauigkeit der Modelle mit Intervallüberlappung in Prozent

6.4. Genauigkeit bei Überlappung der Intervalle

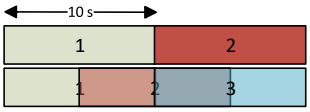


Abbildung 6.5.: Intervallüberlappung

Bao et al. nutzten 2004 auf der Basis bereits bestehender Werke für die Transformation eine Intervallüberlappung von 50% [3]. Weiss et al. probierten dies laut ihrem Paper nicht aus [29]. Um zu prüfen, ob dies eine weitere Verbesserung der Genauigkeiten erzielen könnte, wurden für die Intervalle $(t_0, t_0+10), ((t_0+10)-10\cdot 0.5, (t_0+10)-10\cdot 0.5+10), ...$ Features generiert und wie zuvor evaluiert. Eine Illustration der Überlappung befindet sich in Abbildung 6.5.

Entgegen der Erwartungen verschlechterten sich die persönlichen RF-Modelle wie in Tabelle 6.6 ersichtlich durch die Überlappung um 0.2 Prozentpunkte, während sich die unpersönlichen Modelle sogar um 5.1 Prozentpunkte verschlechterten.

6.5. Genauigkeit bei Verschmelzung der Ess- und Trinkaktivitäten

Betrachtet man die Konfusionsmatrix der vollständigen unpersönlichen RF-basierten Modelle in Tabelle 6.8 auf Seite 38, so lässt sich feststellen, dass die Klassen, die sich mit Essen oder Trinken beschäftigen, nur ungenau erkannt werden und somit die Durchschnittsgenauigkeit entsprechend senken. Zudem liegt die fehlerhaft vorhergesagte Klasse in $298/324 \approx 92\%$ der Fälle im Bereich des Essens und Trinkens. Zurückzuführen ist dies auf die Tatsache, dass sich diese Aktivitäten untereinander ähneln, da die zentralen Bewegungsmerkmale bei all diesen Klassen das Sitzen sowie das Führen der Hand zum Mund sind.

Tabelle 6.7 zeigt, wie sich die Genauigkeit der unpersönlichen Modelle verändert, wenn die Aktivitäten *Eating Chips, Eating Pasta, Eating Sandwich, Eating Soup* und *Drinking* zu einer einzigen Aktivität *Eating* verschmolzen werden. Bei den RF-Modellen ist eine Verbesserung um 8.6 Prozentpunkte zu erkennen, wovon ein Teil auch darauf zurückzuführen sein könnte, dass es insgesamt weniger Klassen gibt und die Wahrscheinlichkeit einer Fehlklassifikation somit sinkt. Tabelle 6.9 auf Seite 39 zeigt die neue Konfusionsmatrix der RF-Modelle und

es wird deutlich, dass aufgrund der verschmolzenen Aktivität *Eating* die Fehlerrate stark gesunken ist.

Aus diesen Daten lässt sich schließen, dass bei der Verwendung von unpersönlichen Modellen darauf geachtet werden sollte, dass die zu erkennenden Aktivitäten nicht zu feingranular voneinander getrennt werden, da dies die Klassifikation stark erschwert.

Algo.	**Comb, Unpers., verschmolzen**	**Comb, Unpers., nicht verschmolzen**
RF	**87.1**	**78.5**
J48	73.7	59.7
IB3	63.9	52.9
NB	72.4	60.9
MLP	77.6	54.9
∅	74.9	61.4

Tabelle 6.7.: Genauigkeit der Modelle bei Verschmelzung der Ess- und Trinkaktivitäten in Prozent

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	< Output für ∨
149	0	0	0	0	0	2	0	0	8	0	0	0	2	0	0	0	0	**a = Brushing Teeth**
0	153	0	7	0	0	0	0	0	0	0	0	0	1	0	0	0	0	**b = Clapping**
0	0	126	0	0	0	0	0	0	0	0	1	17	0	0	0	0	26	**c = Climbing Stairs**
0	11	0	147	0	0	0	0	0	1	0	0	0	5	0	0	0	0	**d = Basketball**
0	0	0	0	133	6	0	9	0	0	0	0	0	0	12	0	3	0	**e = Drinking**
1	0	0	0	5	135	3	15	4	0	0	0	0	0	0	0	1	0	**f = Eating Chips**
3	0	0	0	0	11	106	3	24	0	3	0	0	0	2	0	9	0	**g = Eating Pasta**
3	0	0	0	14	26	15	75	27	0	6	0	0	0	7	1	3	0	**h = Eating Sandwich**
7	0	0	0	0	13	39	44	84	0	0	0	0	0	3	0	2	0	**i = Eating Soup**
6	0	0	0	0	0	0	0	0	148	0	0	3	2	0	0	0	0	**j = Folding Clothes**
0	0	0	0	0	5	7	6	2	0	139	0	0	0	0	0	4	0	**k = Handwriting**
0	0	2	0	0	0	0	0	0	0	0	166	0	0	0	0	0	0	**l = Jogging**
0	0	13	0	0	0	0	0	0	0	0	1	150	0	0	0	0	2	**m = Soccer Ball**
0	0	0	7	0	0	0	0	0	0	0	0	3	145	0	0	0	0	**n = Playing Catch**
4	0	0	0	1	2	2	1	1	0	4	0	0	0	136	8	5	0	**o = Sitting**
1	0	0	0	0	0	0	1	0	0	0	0	1	0	15	142	0	0	**p = Standing**
2	0	0	0	1	1	4	1	0	0	27	0	0	0	0	0	125	0	**q = Typing**
0	0	66	0	0	0	0	0	0	0	0	0	19	0	0	0	0	84	**r = Walking**

Tabelle 6.8.: Konfusionsmatrix der unpersönlichen RF-Modelle

a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**< Output für >**
143	0	0	0	5	12	0	0	0	1	0	0	0	0	**a = Brushing Teeth**
0	155	0	5	0	0	0	0	0	1	0	0	0	0	**b = Clapping**
0	0	126	0	0	0	0	1	13	0	0	0	0	30	**c = Climbing Stairs**
0	5	0	136	0	5	0	0	0	18	0	0	0	0	**d = Basketball**
2	0	0	0	830	0	2	0	0	0	16	0	7	0	**e = Eating**
2	0	0	0	0	155	0	0	0	2	0	0	0	0	**f = Folding Clothes**
0	0	0	0	30	0	124	0	0	0	0	0	9	0	**g = Handwriting**
0	0	0	0	0	0	0	168	0	0	0	0	0	0	**h = Jogging**
0	0	6	0	0	0	0	1	157	1	0	0	0	1	**i = Soccer Ball**
0	0	0	7	0	2	0	1	2	143	0	0	0	0	**j = Playing Catch**
1	0	0	0	23	0	2	0	0	0	130	8	0	0	**k = Sitting**
0	0	0	0	5	0	0	0	0	0	17	138	0	0	**l = Standing**
1	0	0	0	37	0	18	0	0	0	0	0	105	0	**m = Typing**
0	0	60	0	0	0	0	0	26	0	0	0	0	83	**n = Walking**

Tabelle 6.9.: Konfusionsmatrix der unpersönlichen, verschmolzenen RF-Modelle

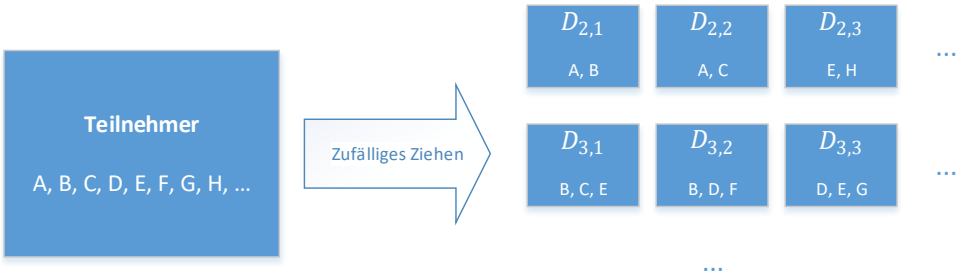


Abbildung 6.6.: Ziehung der Datensätze $D_{i,j}$

6.6. Genauigkeit in Abhängigkeit von der Teilnehmeranzahl

Während die persönlichen Modelle lediglich auf die Bewegungsdaten einer einzelnen Person zurückgreifen, sind für die unpersönlichen Modelle Daten anderer Personen erforderlich. Daraus ergibt sich die Frage, wie die Genauigkeit der unpersönlichen Modelle von der Anzahl der Personen, auf denen dieses basiert, abhängig ist. Um dies zu ermitteln, wurden aus dem aus 10 Personen bestehenden vollständigen Datensatz D für $i \in \{2,...,10\}$ je $N_i := \min(10, \binom{10}{i})$ verschiedene, zufällig gezogene Datensätze $D_{i,j}$ mit $j \in \{1,...,N_i\}$ generiert. Dieser Prozess ist in Abbildung 6.6 illustriert. Ein Datensatz $D_{i,j}$ enthält Daten von $i \geq 2$ verschiedenen Personen, um die Aufteilung in Train- und Testsets zu ermöglichen, da ein Testset für die unpersönlichen Modelle genau eine Person enthalten soll und die Mengen disjunkt sein müssen (vgl. Abschnitt 6.1.2). Die Genauigkeit in Abhängigkeit von der Anzahl der Personen i ist dann definiert durch $G(i) := \sum_{j=1}^{N_i} \frac{\text{Genauigkeit mit } D_{i,j}}{N_i}$. Die Berechnung der Genauigkeit mit Datensatz $D_{i,j}$ erfolgt nach Abschnitt 6.1.2. Abbildung 6.7 zeigt den Plot der Funktion G. Es ist zu erkennen, dass die Genauigkeit langsam konvergiert, wobei für $i > 10$ noch etwas höhere Genauigkeiten zu erwarten sind. Die Verifikation dieser Hypothese durch weitere Aufnahmen war im Rahmen der zeitlichen Beschränkungen einer Bachelorarbeit nicht möglich.

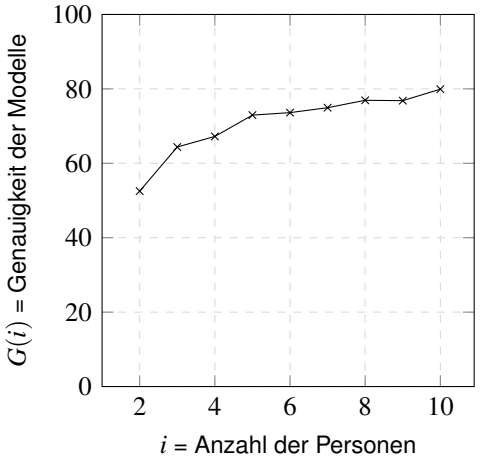


Abbildung 6.7.: Genauigkeit in Abhängigkeit von der Personenanzahl

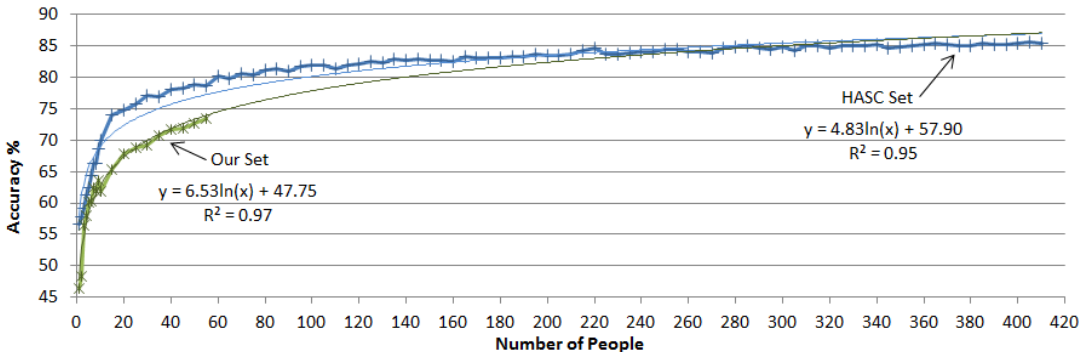


Abbildung 6.8.: Genauigkeit in Abhängigkeit von der Personenzahl (Lockhart & Weiss [18])

Zum Vergleich befindet sich in Abbildung 6.8 eine Grafik aus [18], die zeigt, wie sich die Genauigkeit unpersönlicher RF-Modelle, die nur aus Daten eines Smartphone-Accelerometers gewonnen wurden, in dieser Arbeit in Abhängigkeit von der Personenanzahl verhält. Der große HASC-Datensatz konvergiert gegen eine Genauigkeit von etwa 85%, die laut Extrapolation der Autoren auch mit einem größeren eigenen Datensatz hätten erreicht werden können. Es fällt auf, dass auch der Plot in Abbildung 6.7 gegen einen ähnlichen Grenzwert zu konvergieren scheint, wobei zu beachten ist, dass die beiden externen Datensätze weniger verschiedene Aktivitäten beinhalten, die zudem hinsichtlich ihrer Erkennbarkeit durch ML-Klassifikation variieren könnten. Zusätzlich ist festzustellen, dass die Genauigkeit in Abbildung 6.7 bei 10 Personen bereits etwa 80% beträgt, während es in Abbildung 6.8 nur etwa 63%, respektive 70% sind. Dies deutet darauf hin, dass mehr Sensoren die Notwendigkeit vieler Personen im Datensatz verringern.

6.7. Optimierung der Hyperparameter

Um die unpersönlichen Modelle weiter zu verbessern, wurden die Hyperparameter des RF-Modells optimiert, da dieses Verfahren bis dato das beste Ergebnis geliefert hat. Als Baseline dient das RF1-Modell, das dieselben Hyperparameter besitzt wie das RF-Modell in Abbildung 6.2. Zu bemerken ist, dass die folgenden Genauigkeiten je nach Seed für den Pseudozufallsgenerator um ± 1.5 Prozentpunkte schwanken, weshalb die Genauigkeit des RF1-Modells im hier gezeigten Durchlauf des Tests rund 1.5 Prozentpunkte über der des RF-Modells in Abbildung 6.2 liegt.

Die variierten Hyperparameter sind die folgenden:

- *numIterations* gibt die Anzahl der Entscheidungsbäume im Wald an. Ein zu niedriger Wert kann dafür sorgen, dass die zufällig generierten Bäume den Datensatz zusammen nicht gut genug abdecken können.

- *numFeatures* gibt die Anzahl der Features an, die jeder Entscheidungsbaum bei einem Split betrachtet. Ist der Wert ∞, so werden alle Features untersucht. Wird ein zu hoher Wert verwendet, ist die Entwicklung eines Baums weniger vom Zufall getrieben. Ist der Wert zu niedrig, so sinkt die Wahrscheinlichkeit, dass ein Split an einem Feature

durchgeführt wird, mit Hilfe dessen der *Information Gain* des Splits bezüglich der Entropie der Blattknoten maximal wird. Eine genauere Erläuterung des *Information Gain* ist zu finden in Abschnitt 3.4.1, Machine Learning, Thomas Mitchell [21].

- *maxDepth* bestimmt die maximale Tiefe der Entscheidungsbäume im Wald. Ein niedriger Wert sorgt dafür, dass der Baum sich den Trainingsdaten nicht zu sehr anpasst und *Overfitting* somit vermieden wird. Dabei muss beachtet werden, dass ein zu niedriger Wert wiederum dafür sorgen kann, dass für eine Instanz zu wenige Features betrachtet werden können, um eine genaue Klassifizierung zu ermöglichen.

	numIterations	**numFeatures**	**maxDepth**	**Genauigkeit in Prozent**
RF8	75	f	50	81.229
RF7	100	f	75	80.7925
RF4	100	f	50	80.6917
RF6	150	f	50	80.1545
RF9	100	f	30	80.0537
RF1	50	10	25	79.953
RF2	200	∞	∞	73.5729
RF5	200	∞	50	73.2371
RF3	200	∞	50	72.6998

Tabelle 6.10.: Hyperparameteroptimierung

Tabelle 6.10 zeigt die Genauigkeiten in Abhängigkeit von den Hyperparametern. In einigen Fällen wurde für *numFeatures* der WEKA-Standardwert $f = \lfloor log_2(n-1)+1 \rfloor = \lfloor log_2(208)+1 \rfloor = 8$ verwendet wird, wobei n die Anzahl der Features einer Instanz ist.

Gegenüber der Baseline ist das RF8-Modell das beste, allerdings nur weniger als 1.3 Prozentpunkte besser, sodass die Differenz unter den Veränderungen durch zufallsbedingte Schwankungen liegt. Es ist jedoch festzustellen, dass der automatisch gewählte Wert f für *numFeatures* die besten Ergebnisse erzielt. Eine Erhöhung der maximalen Tiefe auf 50 oder gar 75 erzielt keine nennenswert besseren Ergebnisse als die oben gewählte Tiefe von 25. Auch für die Anzahl der Bäume gilt, dass bereits 50 annähernd so gut sind wie 100 oder mehr. Positiv zu vermerken ist somit, dass in diesem Anwendungsfall von Random Forests auch kleine Werte eine gute Genauigkeit liefern, wobei der Leistungsbedarf entsprechend niedriger ist.

6.8. Einfluss von Feature-Selection

Betrachtet man die *Comb*-Spalte in Tabelle 6.2, so fällt auf, dass die Genauigkeiten der übrigen Modelle dem RF-Modell deutlich unterlegen sind. Da dies bei den persönlichen Modellen nicht der Fall ist, und es somit kein inhärentes Problem mit der Art der Daten zu geben scheint, ist es untersuchenswert, ob die anderen Algorithmen nicht durch eine Vorauswahl der Features (*Feature Selection*) höhere Genauigkeiten liefern können, da eine solche Auswahl prinzipiell auch durch den Zufall im Random Forest erfolgt.

Zur Feature Selection wurde das Verfahren *CfsSubsetEval* [10] verwendet, das laut WEKA-Dokumentation wie folgt arbeitet:

> „[CfsSubsetEval] ermittelt den Wert einer Untermenge der Features, indem die individuelle Vorhersagefähigkeit eines jeden Features mitsamt dem Grad der Redundanz zwischen ihnen betrachtet wird. Feature-Untermengen, die stark mit der Klasse korrelieren und dabei eine niedrige Interkorrelation haben, werden bevorzugt."
>
> *– aus [31] übersetzt.*

Begonnen wurde mit der leeren Menge, der nach und nach Features hinzugefügt worden sind, bis die Bewertung durch *CfsSubsetEval* sank.

Unter Einsatz der Feature Selection entstanden die Ergebnisse aus Tabelle 6.11. Gegenüber Tabelle 6.2 sind die Genauigkeiten der IB3-, NB- und MLP-Modelle wesentlich besser, wodurch auch der Durchschnittswert angehoben wird, jedoch schlägt weiterhin kein Algorithmus den RF-Algorithmus. Diesen hat die Feature Selection des Weiteren nicht verbessert, sodass sich die Feature Selection in diesem Anwendungsszenario als nicht sinnvoll erwiesen hat.

Algo.	**Comb, Unpers.**
RF	78.2
J48	60.0
IB3	59.6
NB	67.8
MLP	68.3
∅	66.8

Tabelle 6.11.: Genauigkeit mit Feature Selection

6.9. Zusatz: Personenerkennung durch Bewegungsdaten

Die von der Transformationssoftware ausgegebenen Daten enthalten neben den regulären Features auch eine Spalte, die den Namen des Teilnehmers angibt. Vor der weiteren Auswertung wurde dieses Feature stets aus dem Datensatz entfernt, jedoch lässt es sich auch zu einem nützlichen Test verwenden. Unter geringem Aufwand konnte geprüft werden, inwiefern die Kombination mehrerer Sensoren für die Personenerkennung anhand von Bewegungsdaten geeignet ist. Diese Idee ist nicht neu: 2010 untersuchten auch Kwapisz et al. auf Basis ihrer bisherigen Entwicklungen bezüglich Aktivitätenerkennung durch Smartphones, ob diese biometrische Identifikation ermöglichen [16] und kamen zu einem positiven Ergebnis.

Nach Entfernung der Aktivität als ehemaliges Target aus den Daten wurde WEKA eingesetzt, um mit Hilfe des auch für die Aktivitätenerkennung verwendeten RF-Modells als Target nun den Namen des Teilnehmers vorherzusagen. Evaluiert wurde die Genauigkeit über eine 10-fache Kreuzvalidierung, dessen Ergebnisse in Tabelle 6.12 zu sehen sind. Mit einer Genauigkeit von 97.7% ist das *Comb*-Modell erwarteterweise am besten. Interessant ist der Vorsprung gegenüber den anderen Modellen: Gegenüber der Kombination der Daten

Datensatz	Genauigkeit in Prozent
Comb	97.7
Band comb	87.3
Band gyro	40.5
Band accel	62.2
Phone comb	85.4
Phone gyro	48.4
Phone accel	83.3

Tabelle 6.12.: Genauigkeit der Personenerkennung

des Band und der Kombination der Daten des Smartphones konnte die Genauigkeit um 10.4 respektive 12.3 Prozentpunkte gesteigert werden. Relativ schwach sind die Modelle, die lediglich einen einzelnen Sensor nutzen, wobei der Beschleunigungssensor des Smartphones eine überraschend gute Erkennungsrate erzielen lässt.

Bei der Interpretation dieser Genauigkeiten ist Vorsicht geboten, da insbesondere falschpositive Erkennungen einer Person zumindest im Kontext der Authentifizierung fatal sind. Da jedoch unter anderem Kwapisz et al. bereits vielversprechende Resultate hinsichtlich dieses Aspekts lieferten, sind die hier erzielten Genauigkeiten ein guter Indikator dafür, dass auch hier die Kombination mehrerer Datenquellen förderlich sein kann, wenn die Verwendung einer einzelnen Quelle keine akzeptablen Genauigkeiten liefert.

7. Fazit

7.1. Zusammenfassung

Die in Abschnitt 1.2 aufgeführten Ziele wurden erreicht. Auf Basis von maschinellem Lernen wurde ein Verfahren entwickelt und implementiert, das mit Hilfe von Sensordaten eines Fitness-Trackers und eines Smartphones körperliche Aktivitäten erkennt und dabei nicht auf eine feste Ausrichtung der Geräte angewiesen ist. Zur Evaluation wurde ein Experiment mit 10 Teilnehmern durchgeführt.

Der erste Teil der Entwicklung widmete sich der Implementierung einer robusten Aufzeichnungssoftware für das Android-Betriebssystem, die Rohdaten simultan von mehreren Sensoren anfordert und in einem definierten Dateiformat in Form von *Readings* aufzeichnet. Eine weitere entwickelte Software transformiert die *Readings* in mit Hilfe von Unterteilung in Intervalle Instanzen, die als Trainings- und Testdaten für konventionelle ML-Algorithmen dienen, die anschließend eingesetzt werden.

Evaluiert wurden die Genauigkeiten von Modellen, die auf Basis verschiedener ML-Algorithmen und mit Hilfe von Datensätzen verschiedener Sensoren entstanden sind. Die wichtigste Fragestellung dieser Arbeit war, ob die Kombination der Daten eines Smartphones und eines Fitness-Trackers die Genauigkeit der Modelle steigern kann. Insbesondere Abschnitt 6.1 beantwortet diese Frage: Persönliche Modelle, die für jeden Teilnehmer exklusiv mit dessen Daten gebildet wurden, erreichen durch die Kombination der Sensoren eine Genauigkeit von 99.4% und sind damit 7.8 Prozentpunkte besser als persönliche Modelle, die lediglich die Daten des Beschleunigungssensors des Fitness-Trackers genutzt haben. Selbst wenn nur die diversen Sensoren des Fitness-Trackers kombiniert werden, kann noch eine Genauigkeit von 97.1% erzielt werden. Unpersönliche Modelle, die für einen Teilnehmer exklusiv nur die Daten der anderen Teilnehmer verwenden, profitieren ebenfalls von der Datenkombination. Durch diese wird eine Genauigkeit von 78.5% erreicht, sodass die Genauigkeit der Modelle auf Basis der besten einzelnen Datenquelle, dem Beschleunigungssensor des Fitness-Trackers, um 3.3 Prozentpunkte übertroffen wird.

Da sich bei der Verwendung unterschiedlicher Sensoren, die in separaten Geräten verbaut sind, die Frage ergibt, inwiefern die Synchronität der *Readings* eine Rolle spielt, wurde der Effekt gaußschen Rauschens auf die Zeitstempel dieser geprüft. Dabei konnte festgestellt werden, dass $\mathcal{N}(0, 500ms)$-verteilte Abweichungen der Zeitstempel der *Readings* keinen nennenswerten Einfluss auf die Genauigkeit haben.

Als weiterer Einfluss auf die Erkennungsrate wurde überprüft, welche Abtastraten für die Aktivitätenerkennung erforderlich sind. Motiviert wurde diese Prüfung dadurch, dass eine höhere Abtastrate zu einem höheren Energieverbrauch führt und die verwendeten Geräte typischerweise keine permanente Stromversorgung haben. In vielen verwandten Arbeiten werden die Sensoren mit 20 Hz abgetastet, jedoch zeigt die Untersuchung, dass für persönliche Modelle sogar eine Abtastrate von 1 Hz ausreichen kann. Unpersönliche Modelle werden

von der Reduzierung der Abtastrate stärker geschwächt, jedoch ist mit 5 Hz immer noch eine Genauigkeit von 74.6% möglich.

Bei der Analyse der Konfusionsmatrizen ist aufgefallen, dass die Ess- und Trinkaktivitäten außergewöhnlich oft untereinander verwechselt werden. Eine Verschmelzung dieser Aktivitäten ermöglichte die Steigerung der Genauigkeit unpersönlicher Modelle um 8.6 Prozentpunkte auf wesentlich praxistauglichere 87.1%. Hieraus konnte die Erkenntnis abgeleitet werden, dass Aktivitäten für unpersönliche Modelle nicht zu feingranular voneinander abgegrenzt werden sollten.

Hinsichtlich der für die Bildung unpersönlicher Modelle erforderlichen Trainingsdaten konnte in Abschnitt 6.6 festgestellt werden, dass die Kombination unterschiedlicher Sensoren die Notwendigkeit vieler Personen im Datensatz verringern kann.

Da der von der Transformationssoftware ausgebene Datensatz hochdimensional ist, wurde der Einfluss von Feature-Selection geprüft, dabei jedoch festgestellt, dass dadurch keine Verbesserungen der Genauigkeit möglich waren.

Zusätzlich zu den eigentlichen Zielen dieser Arbeit konnte außerdem herausgefunden werden, dass auch Personen- statt Aktivitätenerkennung hinsichtlich der Erkennungsrate von der Kombination diverser Sensoren profitieren kann.

7.2. Beurteilung der Ergebnisse

Diese Arbeit zeigt, dass die Kombination mehrerer Sensoren zur Gewinnung von ML-Modellen konsistent bessere Ergebnisse liefert als die Verwendung einzelner Sensoren. Insbesondere persönliche Modelle profitieren hiervon, da eine fast perfekte Erkennungsrate erreicht wird. Die in dieser Arbeit verwendete Transformationsmethode, die Features aus den *Readings* extrahiert, ist gegenüber der Desynchronisierung der Zeitstempel und unterschiedlichen Ausrichtungen der Geräte resistent, weshalb Anwendern diesbezüglich keine praxisuntauglichen Einschränkungen auferlegt werden müssen. Werden unpersönliche Modelle verwendet, sollte darauf geachtet werden, dass sich die definierten Aktivitäten hinreichend voneinander unterscheiden, da beispielsweise das Auseinanderhalten von Essaktivitäten schwierig ist.

Um die Schwäche unpersönlicher Modelle auszugleichen, könnte eine Anwendung in der Praxis für einen neuen Benutzer zunächst auf ein rein-unpersönliches Modell setzen, das dessen Aktivitäten klassifiziert. Die Anwendung kann den Nutzer um Rückmeldungen bitten, ob die vorhergesagte Klasse korrekt war und mit diesen Daten ein hybrides Modell bauen, das sowohl persönliche als auch unpersönliche Daten verwendet. Laut Weiss et al. wird ein persönliches Modell durch die Rückmeldungen des Benutzers nach einiger Zeit genauer als das hybride Modell, sodass auf dieses umgestiegen werden kann [30], sofern alle klassifizierbaren Aktivitäten des unpersönlichen Modells auch durch das persönliche Modell abgedeckt werden.

7.3. Ausblick

In Zukunft sind insbesondere hinsichtlich der Transformation der Daten Weiterentwicklungen denkbar. Einerseits könnten die Aufnahmen verschiedener Sensoren noch intensiver genutzt werden, indem beispielsweise Korrelationsfeatures zwischen den Komponenten der Sensoren hinzugefügt werden und andererseits könnten die Features für mehrere aufeinanderfolgende

Intervalle berechnet und als eine Instanz verwendet werden, um die Sequentialität der Signale besser in den Trainingsdaten wiederzugeben. Aufgrund der höheren Dimensionalität des daraus resultierenden Datensatzes müssten dafür allerdings mehr Daten aufgenommen werden.

Des Weiteren könnten verschiedene Intervallgrößen ausprobiert werden, anstatt diese auf 10 Sekunden festzulegen. Auch intelligentere Intervallbildungsmethoden, die Aspekte wie Periodizität in den aufgenommenen Daten erkennen, sind denkbar.

Da es sich bei WEKA um eine Java-basierte Software handelt, könnten sowohl das Training als auch die Klassifizierung mittels eines fertigen Modells auf dem Android-Gerät selbst realisiert werden, da Anwendungen für dieses Betriebssystem ebenfalls in Java geschrieben werden. Im Test an einem stationären Computer war das Training eines RF-Modells in der Regel innerhalb weniger Sekunden abgeschlossen, sodass auch mobile Prozessoren in der Lage dazu wären, das Training in kurzer Zeit auszuführen. Dies trifft auch auf die Klassifizierung selbst zu, da diese neben der auch für das Training erforderlichen Transformation der Intervalle lediglich das Durchlaufen einer festen Anzahl von Entscheidungsbäumen erfordert.

Nach der Einreichung dieser Bachelorarbeit wird der aufgenommene Datensatz anonymisiert und auf meiner Webseite (`https://cbruegg.com`) zur Verfügung gestellt werden.

7.4. Reflexion

Für die Aufzeichnung der *Readings* wurde der Einfachheit wegen JSON-Serialisierung verwendet, die sich jedoch im Laufe der Arbeit als ineffizient herausstellte. Aufgrund der Menge der Messdaten mussten diese zusätzlich mit GZIP komprimiert werden, um in kurzer Zeit vom Smartphone auf den Datenspeicher hochgeladen werden zu können. Die Deserialisierung des gesamten Datensatzes, der komprimiert noch eine Größe von 231 MiB hat, dauerte innerhalb der Transformationssoftware bis zu einer Minute und sorgte somit bei Tests dieser Software für Verzögerungen. Stattdessen empfehlenswert wäre im Nachhinein die Verwendung eines binären Dateiformats gewesen.

Als eine gute Entscheidung hingegen hat sich erwiesen, alle Rohdaten unbearbeitet zu speichern und durch eine separate Software zu transformieren, anstatt dies in einem Schritt in der Aufzeichnungssoftware zu tun. So war es möglich, viele Parameter auch nachträglich anzupassen.

Anhang

A. Verwendete Akronyme

ML Machine Learning

RF Random Forest

IB Instance-Based-Learner (IBk)

J48 J48-Entscheidungsbaum

NB Naive Bayes

MLP Multilayer Perceptron

NTP Network Time Protocol

SDK Software Development Kit

HASC Human Activity Sensor Consortium

JSON JavaScript Object Notation

GZIP GNU Zip

B. Literaturverzeichnis

[1] D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] Daniel Ashbrook. Context sensing with the twiddler keyboard. In *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*, pages 197–198. IEEE, 1999.

[3] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *International Conference on Pervasive Computing*, pages 1–17. Springer, 2004.

[4] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[5] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] JBJ Bussmann, WLJ Martens, JHM Tulen, FC Schasfoort, HJG Van Den Berg-Emons, and HJ Stam. Measuring daily behavior using ambulatory accelerometry: the activity monitor. *Behavior Research Methods, Instruments, & Computers*, 33(3):349–356, 2001.

[7] S. Dernbach, B. Das, N. C. Krishnan, B. L. Thomas, and D. J. Cook. Simple and complex activity recognition through smart phones. In *Proc. Eighth Int. Conf. Intelligent Environments*, pages 214–221, June 2012.

[8] Jonny Farringdon, Andrew J Moore, Nancy Tilbury, James Church, and Pieter D Biemond. Wearable sensor badge and sensor jacket for context awareness. In *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*, pages 107–113. IEEE, 1999.

[9] Fitbit. What should I know about SmartTrack exercise detection? https://help.fitbit.com/articles/en_US/Help_article/1933. Abgerufen am 19.02.2017.

[10] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[11] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[12] Simon Haykin. Neural networks, a comprehensive foundation. -, 1994.

[13] Google Inc. *Android Sensors Overview*. Google Inc., 12 2016.

[14] George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, San Mateo, 1995. Morgan Kaufmann.

[15] A. Krause, M. Ihmig, E. Rankin, D. Leong, Smriti Gupta, D. Siewiorek, A. Smailagic, M. Deisher, and U. Sengupta. Trading off prediction accuracy and power consumption for context-aware wearable computing. In *Proc. Ninth IEEE Int. Symp. Wearable Computers (ISWC'05)*, pages 20–26, October 2005.

[16] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Cell phone-based biometric identification. In *Proc. Applications and Systems (BTAS) 2010 Fourth IEEE Int. Conf. Biometrics: Theory*, pages 1–7, September 2010.

[17] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.*, 12(2):74–82, March 2011.

[18] Jeffrey W Lockhart and Gary M Weiss. The benefits of personalized smartphone-based activity recognition models. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 614–622. SIAM, 2014.

[19] IC McManus, James Moore, Matthew Freegard, and Richard Rawles. Science in the making: Right hand, left hand. iii: Estimating historical rates of left-handedness. *Laterality*, 15(1-2):186–208, 2010.

[20] D Mills, J Martin, J Burbank, and W Kasch. Rfc5905. *Network Time Protocol Version*, 4:2010–06, 2010.

[21] Thomas Mitchell. *Machine Learning*. McGraw-Hill Education - Europe, 1997.

[22] „Moxfyre". Aliasing sine. https://en.wikipedia.org/wiki/File:AliasingSines.svg, 04 2009.

[23] AY Ng. Cs229 lecture notes on machine learning. Technical report, Technical report, Stanford University, Department of Computer Science, Stanford, USA, 2011. 39, 116, 140, 2011.

[24] Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[25] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L Littman. Activity recognition from accelerometer data. In *Aaai*, volume 5, pages 1541–1546, 2005.

[26] Albrecht Schmidt, Michael Beigl, and Hans-W Gellersen. There is more to context than location. *Computers & Graphics*, 23(6):893–901, 1999. Ursprünglich erschienen im Nov. 1998.

[27] B. N. Taylor and A. Thompson. The international system of units (si). National Institute of Standards and Technology, Special Publication 330, Gaithersburg, MD, 2008.

[28] Kristof Van Laerhoven and Ozan Cakmakci. What shall we teach our pants? In *Wearable Computers, The Fourth International Symposium on*, pages 77–83. IEEE, 2000.

[29] G. M. Weiss, J. L. Timko, C. M. Gallagher, K. Yoneda, and A. J. Schreiber. Smartwatch-based activity recognition: A machine learning approach. In *Proc. IEEE-EMBS Int. Conf. Biomedical and Health Informatics (BHI)*, pages 426–429, February 2016.

[30] Gary M Weiss and Jeffrey W Lockhart. The impact of personalization on smartphone-based activity recognition. In *AAAI Workshop on Activity Context Representation: Techniques and Languages*, pages 98–104, 2012.

[31] WEKA. WEKA Javadoc: CfsSubsetEval, 02 2017.

Erklärung der Urheberschaft

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Ort, Datum

Unterschrift