

Aktivitätserkennung auf Smartphones

Tobias Wirtl, Claudia Nickel

Hochschule Darmstadt

tobias.wirtl@stud.h-da.de, claudia.nickel@h-da.de

Abstract: Mobile Geräte wie Handys haben seit Jahren einen festen Platz in unserem Alltag. In den letzten Jahren hat sich auf diesem Sektor daher viel verändert und eine neue Geräteklasse, die sogenannten Smartphones, halten Einzug in den Massenmarkt. Da die Geräte nicht mehr nur zum Telefonieren genutzt werden, sind in einem modernen Smartphone allerlei Sensoren integriert. Unter anderem auch Beschleunigungsmesser, die beispielsweise für die Gangerkennung genutzt werden können. Dieser Artikel beschäftigt sich mit einer Ergänzung zur Verbesserung der Gangerkennung, es wird eine Möglichkeit geschaffen, über Beschleunigungssensoren in Smartphones verschiedene Aktivitäten zu unterscheiden, die der Benutzer während des Tragens des Geräts ausführt. Im Gegensatz zu anderen Arbeiten wurde hierbei die Erkennung direkt auf dem Endgerät implementiert. Es konnte eine Möglichkeit gefunden werden, die Aktivitätserkennung auf einem Smartphone so zu realisieren, dass die benötigte Zeit zur Erkennung, sowie die Zuverlässigkeit im akzeptablen Bereich liegen.

1 Einleitung

In den letzten Jahren hat sich die Rolle des Mobiltelefons stark verändert. Die Geräte werden nicht mehr nur zum Telefonieren und Nachrichten schreiben verwendet, sondern dienen auch der Unterhaltung. Ein Nebeneffekt dieser Entwicklung ist die Integration diverser Sensoren in die Geräte. Besonders die Klasse der Smartphones besitzt hierbei eine große Auswahl. Neben Sensoren zum Messen von Temperatur, Licht, Schwerkraft, Luftdruck etc., befindet sich fast in jedem dieser Geräte ein 3-Achsen-Beschleunigungsmesser. Andere Projekte haben gezeigt, dass es möglich ist, mit solchen Sensoren beispielsweise eine Gangerkennung zu realisieren [NDBB11]. Ein Problem hierbei ist, dass zunächst erkannt werden muss, ob der Benutzer überhaupt läuft oder eine andere Aktivität ausführt. Diese Arbeit beschäftigt sich mit genau dieser Aktivitätserkennung. Es wird im Folgenden besonders auf die Aktivitäten schnelles, normales und langsames Gehen eingegangen und versucht, diese von anderen Aktivitäten zu unterscheiden. Durch die Berücksichtigung der Klassifikationsergebnisse in der Gangerkennung (z.B. bei der Parameterwahl) kann diese weiter verbessert werden.

Als Ausgangspunkt diente eine Arbeit der Fordham University [KWM10]. Hier wurden verschiedene Aktivitäten (laufen, joggen, treppensteigen (auf/ab), sitzen und stehen) mit den Beschleunigungssensoren verschiedener mobiler Geräte erfasst. Im Gegensatz zu unserem Ziel, die Daten direkt auf dem Gerät auszuwerten, wurden die Daten nach der Erfassung allerdings mit Hilfe anderer Computer ausgewertet. Da auf einem mobilen Gerät

weniger Rechenleistung zur Verfügung steht, muss es der Algorithmus zur Klassifizierung ermöglichen, schnell eine Aktivitätserkennung durchzuführen, auch wenn nur geringe Hardware-Ressourcen verfügbar sind.

2 Umfeld und Werkzeuge

Zur Realisierung stehen zwei Android-basierte Smartphones zur Verfügung. Zum einen ein HTC G1 (2008) und zum anderen ein Motorola Defy (2010). Das HTC G1 hat eine durchschnittliche Samplerate von 43Hz, das Motorola Defy erreicht 137Hz. Beide Geräte verwenden unterschiedliche Versionen des Android Betriebssystems. Das etwas ältere G1 verwendet die Version 1.6, während das neuere Defy die aktuellere Version 2.1 nutzt.



Abbildung 1: Trageposition.

Zur Klassifikation der Aktivitäten wurde die Data Mining Software „Weka“ der Universität von Waikato [wek11] ausgewählt. Die Quellen wurden basierend auf der bestehenden Android Bibliothek [and11] auf Android 1.6 portiert.

Für die Untersuchung standen 151 Datensätze zur Verfügung: 51 x schnelles Gehen, 49 x normales Gehen und 51 x langsames Gehen. Die Daten wurden von 51 Probanden mit dem HTC G1 auf einer geraden Strecke mit ebenem Untergrund erfasst. Die Erfassungsdauer liegt bei ca. 17 Sekunden. Dabei wurde das Telefon waagrecht am Gürtel der Probanden getragen (siehe Abbildung 1).

3 Merkmale

Um eine Verarbeitung der erfassten Daten zu ermöglichen, wurden sechs Merkmale herangezogen, die auch bei ähnlichen Projekten angewendet wurden [KWM10]. Es handelt sich hierbei hauptsächlich um Werte, die zum Gebiet der beschreibenden Statistik gehören. Bis auf das Merkmal „durchschnittliche resultierende Beschleunigung“, werden alle Merkmale getrennt für die jeweilige Sensorachse (x , y , z) ermittelt.

Da die Android API bei der Erfassung keine feste Samplerate ermöglicht, wurden die Daten vor der Merkmalsextraktion interpoliert. Hierbei kamen zwei Verfahren zum Einsatz: Die Übernahme des nächstgelegenen Wertes sowie eine lineare Interpolation der Werte. Die Auswirkungen werden in Abschnitt 4 behandelt.

- **Zeit zwischen Extremwerten (TBP)**

Hierbei werden zunächst alle Extremwerte der erfassten Werte ermittelt. Dies geschieht mit Hilfe der Berechnung der Steigung. Ein Wechsel von positiver zu negativer Steigung wird als Maximum, ein Wechsel von negativer zu positiver Steigung wird als Minimum vermerkt.

Nach der Erfassung aller Maxima (A) und Minima (I) wird die absolute Differenz von $A_{\max}$ und $I_{\min}$ berechnet, woraus ein Grenzwert zur Auffindung interessierender negativer bzw. positiver Extremwerte bestimmt wird.

$$\begin{aligned} g &= 5\% \cdot (A_{\max} - I_{\min}) \\ \text{Grenzwert}_{\max} &= A_{\max} - g \\ \text{Grenzwert}_{\min} &= I_{\min} + g \end{aligned}$$

Extremwerte über der Maxima-Grenze und Extremwerte unter der Minima-Grenze werden getrennt gespeichert. Findet man hierbei jeweils weniger als drei Punkte, wird die Grenze um g verkleinert bzw. vergrößert. Wenn mindestens drei Werte gefunden werden können, wird ein Mittelwert der zeitlichen Abstände der Punkte gebildet. Dies geschieht pro Achse getrennt für Maxima und Minima.

- **Mittelwert (AVG)** getrennt für alle Achsen ($a \in x, y, z$)

$$AVG = \bar{a} = \frac{1}{n} \sum_{i=1}^n a_i \quad (1)$$

- **Standardabweichung (STD)** getrennt für alle Achsen

$$STD = s = \sqrt{\frac{1}{n} \sum_{i=1}^n (a_i - \bar{a})^2} \quad (2)$$

- **Durchschnittliche absolute Differenz (AAD)** getrennt für alle Achsen

$$AAD = \frac{1}{n} \sum_{i=1}^n |a_i - \bar{a}| \quad (3)$$

- **Durchschnittliche resultierende Beschleunigung (ARA)**

$$ARA = \frac{1}{n} \sum_{i=1}^n \sqrt{x_i^2 + y_i^2 + z_i^2} \quad (4)$$

- **Klasseneinteilung (BIN)**

Für jede Achse wird die Spanne ($A_{\max} - I_{\min}$) ihrer Werte in zehn gleichgroße Klassen geteilt. Anschließend wird gezählt, wie viele der Werte der jeweiligen Achse den einzelnen Klassen zugeordnet werden können.

4 Untersuchung

Um die Klassifizierung von Aktivitäten möglichst effizient zu realisieren, wurde zunächst eine Untersuchung mit den vorhandenen Testdaten auf einem PC-System durchgeführt. Hierbei wurden die Parameter *Klassifizierungsalgorithmus*, *Merkmale*, *interpolierte Samplerate*, *Dauer der Datenerfassung* und *Art der Interpolation* herangezogen. Um zu sehen, ob sich die Aufbereitung der erfassten Daten überhaupt positiv auswirkt, wurden auch Ergebnisse ohne Vorverarbeitung, d.h. erfasste Rohdaten variabler Erfassungsdauer und ohne feste Samplerate untersucht.

Für alle Tests wurden zwei Drittel (34 x schnelles Gehen, 32 x normales Gehen, 34 x langsames Gehen) der Daten zum trainieren des Klassifizierungsalgorithmus verwendet. Anhand des restlichen Drittels (17 x schnelles Gehen, 17 x normales Gehen, 17 x langsames Gehen) wurde die Genauigkeit der Algorithmen überprüft.

Es wurden alle Kombinationen der folgenden Parameter getestet:

- Klassifizierungsalgorithmus (18 Algorithmen)
- Alle Merkmalskombinationen
- Interpolierte Samplerate (50Hz, 100Hz, 150Hz)
- Dauer der Datenerfassung (2s, 5s, 10s, 15s)
- Interpolation (nächstgelegener Wert, linear)

Hieraus resultieren $27216 (= 18 \cdot (2^6 - 1) \cdot 3 \cdot 4 \cdot 2)$ Einzeltests. Die gewählten Klassifizierungsalgorithmen entsprechen den auf Android 1.6 lauffähigen Algorithmen der Weka Data Mining-Software. Es wurden die Standardparameter verwendet.

Nach der Datenerfassung wurde zunächst die Grenze der geforderten minimalen Erfolgsrate bei der Klassifizierung auf 95% festgelegt. Die Erfolgsrate entspricht hierbei richtig klassifizierten Datensätzen innerhalb eines Tests. Der Algorithmus darf also um diese Rate zu erreichen, maximal zwei der 51 Testdatensätze falsch klassifizieren. Innerhalb dieser Grenze stellten sich zwei Algorithmen als besonders effizient heraus und konnten bei vielen Tests die geforderte Erfolgsrate erreichen (siehe Tabelle 1). Zum einen SimpleLogistic, der auf logistischer Regression basiert, zum anderen der REPTree Algorithmus, der Entscheidungsbäume nutzt. Bei beiden Algorithmen konnte zudem eine hohe durchschnittliche Erfolgsrate innerhalb aller Tests gemessen werden (siehe Tabelle 1).

	Anzahl der Tests mit einer Erfolgsrate > 95%	Durchschnittliche Erfolgsrate
REPTree	296	82,16%
SimpleLogistic	215	82,19%
LMT	194	82,23%
...	...	...
Gesamt/Mittelwert	1287 (von 27216)	75,74%

Tabelle 1: Untersuchung verschiedener Weka Klassifizierungsalgorithmen.

Bei der weiteren Untersuchung dieser zwei Algorithmen wurde die Auswirkung der Änderung von Erfassungsdauer und interpolierter Samplerate näher betrachtet. Sowohl bei REPTree als auch bei SimpleLogistic konnten die besten Ergebnisse bei einer Erfassungsdauer von zehn Sekunden erreicht werden. Die Änderung der Samplerate wirkte sich unterschiedlich auf die Algorithmen aus. Bei REPTree war das Ergebnis für alle Frequenzen gleich, bei SimpleLogistic hingegen gab es große Unterschiede (siehe Tabelle 2). Ein Vergleich mit der Verarbeitung der Rohdaten zeigt, dass die Vorverarbeitung (Interpolation) besonders bei REPTree wesentliche Vorteile bringt. Ohne Vorverarbeitung gab es hier keine Tests mit Erfolgsraten über 95%, bei SimpleLogistic war das Ergebnis mit 25 Tests in einem guten Bereich.

	nächstgelegener Wert		lineare Interpolation	
	REPTree	SimpleLogistic	REPTree	SimpleLogistic
50Hz	32	13	32	16
100Hz	32	8	32	9
150Hz	32	22	32	30

Tabelle 2: Tests mit einer Erfolgsrate über 95% bei einer Erfassungsdauer von 10 Sekunden

Aufgrund der vorliegenden Ergebnisse wurde im weiteren der REPTree Algorithmus verwendet. Bezüglich der Auswirkung der Kombination von Merkmalen konnten kaum Unterschiede festgestellt werden. Die Erkennung scheint hauptsächlich vom Merkmal STD abhängig zu sein. Jede Kombination dieses Merkmals mit anderen ergab eine Erkennungsleistung von über 95% innerhalb der Testdaten. Ein Blick auf den resultierenden Suchbaum, der unter Verwendung aller Merkmale entstand, zeigt den Grund hierfür. Der Algorithmus hat lediglich die Merkmale „Standardabweichung der x-Achse“ und die „durchschnittliche absolute Differenz der x-Achse“ zur Klassifizierung verwendet.

5 Implementierung auf dem Smartphone

Mit Hilfe der gewonnenen Erkenntnisse wurde eine Applikation erstellt, die es ermöglicht, ohne weitere Hardware Daten zu erfassen und diese mit der Weka-Bibliothek zu klassifi-

zieren. Zur Klassifikation wird hierbei der REPTree Algorithmus sowie alle in 3 genannten Merkmale eingesetzt. Die Erfassungsdauer liegt bei 10 Sekunden, die Samplerate bei 100Hz. Dies entspricht einem Upsampling auf dem HTC G1, sowie einem Downsampling auf dem Motorola Defy.

Eine Erfassung neuer Daten wurde ebenfalls integriert. Hierzu können neue Aktivitäten angelegt und erlernt werden. Da es durch die Datenerfassung notwendig ist, den Klassifizierer neu zu erstellen, ist auch diese Funktion implementiert. Sowohl auf dem Motorola Defy, als auch auf dem HTC G1 ist die Erstellung des REPTree-Klassifizierers in akzeptabler Zeit möglich. Hierzu werden zunächst die auf dem Handy gespeicherten Datensätze geladen und deren Merkmale extrahiert. Mit diesen Daten kann der eigentliche Klassifizierer erstellt werden (Lernphase) (siehe Tabelle 3). Auffällig ist die Dauer zum Einlesen der Daten. Hierbei wird jeder Datensatz aus einer eigenen Datei gelesen, die auf der SD-Karte des Smartphones gespeichert ist. Um diese Zeit zu verkürzen sollte in späteren Versionen eine Datenbank eingesetzt oder alle Datensätze in eine Datei gespeichert werden.

Aktion	Defy	G1
Einlesen der gespeicherten Datensätze	149,73s	648,18s
Extraktion der Merkmale aus den eingelesenen Datensätzen	6,87s	30,57s
Lernphase zur Erstellung des Klassifizierers	1,76s	9,35s
Gesamt	156,89s	679,54s

Tabelle 3: Benötigte Zeiten zur Erstellung des Klassifizierers bei 151 Datensätzen.

Die Klassifizierung erfasster Datensätze mit Hilfe des erstellten Klassifizierers ist in 0,29s (Motorola Defy) bzw. 0,79s (HTC G1) möglich. Ein Test zur Erstellung eines Klassifizierers mit Hilfe des MultilayerPerceptron-Algorithmus zeigte, dass die mit REPTree erreichten Ergebnisse nicht für alle Algorithmen möglich sind. Die Lernphase benötigte hier beispielsweise über neun Minuten auf dem Motorola Defy.



Abbildung 2: Benutzeroberfläche der Anwendung.

6 Evaluation

Die erstellte Applikation wurde mit verschiedenen Probanden getestet. Hierbei wurde der Klassifizierer mit allen zur Verfügung stehenden Datensätzen für schnelles, normales und langsames gehen trainiert und zudem eine weitere Aktivität angelegt, die es ermöglichen soll, gehen von anderen Aktivitäten zu unterscheiden. Da bisher keine Daten für diese „keine“-Aktivität zur Verfügung standen, wurde jeweils ein Beispiel für Stehen, „in der Hand halten“, „in der Hand halten und Touchscreen verwenden“, „abgelegtes Handy mit Display oben“, „abgelegtes Handy mit Display unten“ von einer Person erfasst.

Soll				Defy	
l	n	s	k	Ist	
7	0	0	0		l
3	7	0	0		n
0	3	10	0		s
0	0	0	0		k

Soll				G1	
l	n	s	k	Ist	
10	4	0	0		l
0	5	3	0		n
0	1	7	0		s
0	0	0	0		k

Soll			Einzeltest	
l	n	s	Defy	
15	1	1	l	Ist
0	14	4	n	
0	0	10	s	

Tabelle 4: Ergebnisse der Evaluation(l=langsam, n=normal, s=schnell, k=keine).

Getestet wurde mit 20 Probanden. Bei 10 Erfassungen wurde das HTC G1 mit einer Samplerate von 43Hz verwendet, bei den restlichen 10 das Motorola Defy mit einer Samplerate von 137Hz. Das Upsampling bzw. Downsampling wurde auf 100Hz festgelegt, die Erfassungsdauer betrug 10 Sekunden (siehe Abschnitt 5). Jeder Proband lief eine gerade Strecke auf ebenem Boden, jeweils einmal langsam, normal und schnell.

Die Erkennungsleistung lag bei 80% (Motorola Defy) bzw. 73% (HTC G1) (siehe Tabelle 4). Die Abweichung von *schnell* ⇔ *normal* oder *normal* ⇔ *langsam* kann durch die von Person zu Person unterschiedliche Auffassung der Geschwindigkeiten erklärt werden. Eine starke Abweichung von *schnell* ⇔ *langsam* kam innerhalb des Tests nicht vor (siehe Tabelle 4).

Ein weiterer Test sollte zeigen, ob eine Erkennung der Aktivitäten *schnelles*, *normales* und *langsames* Gehen durch wenige Daten einer Testperson möglich ist. Hierzu wurden 12 Datensätze erfasst (4 x schnell, 4 x normal, 4 x langsam) und hieraus ein Klassifizierer erstellt. In 45 Läufen (15 x schnell, 15 x normal, 15 x langsam) konnte eine Erkennungsleistung von etwa 86% gemessen werden (siehe Tabelle 4). Auffällig war die schlechte Erkennung des schnellen Gehens, wobei dieses sogar als *langsam* gewertet wurde.

7 Fazit

Es konnte gezeigt werden, dass eine Erkennung der Aktivitäten *langsames*, *normales* und *schnelles* Gehen auf einem Smartphone realisiert werden kann. Selbst auf älteren Geräten, wie dem HTC G1 stehen genügend Hardwareressourcen zur Verfügung.

Bei Feldversuchen war es möglich, Erkennungsraten von über 80% zu erreichen. Ein Vergleich mit den Ergebnissen anderer Untersuchungen zum Thema Aktivitätserkennung [KWM10] zeigt, dass diese Erkennungsraten trotz der nahen Verwandtschaft der untersuchten Aktivitäten (langsames, schnelles und normales Gehen), bereits in einem sehr guten Bereich liegen. Diese Ergebnisse sowie die kurzen Klassifizierungszeiten zeigen, dass das beschriebene Verfahren in der biometrischen Gangerkennung als Vorverarbeitungsschritt eingesetzt werden kann. Verschiedene Geschwindigkeiten können identifiziert werden, um ein für die entsprechende Geschwindigkeit optimiertes Gangerkennungsverfahren zur Authentisierung auszuwählen.

In Zukunft muss die Auswirkung der Aufnahme anderer Aktivitäten auf die Erkennungsleistung untersucht werden. Zudem müssen Daten gesammelt werden, die es ermöglichen die interessierenden Aktivitäten von anderen zu unterscheiden. Die erstellte Applikation, sowie die portierte Weka-Bibliothek bilden hierfür bereits eine sehr gute Grundlage.

Literatur

- [and11] <https://github.com/rjmarsan/Weka-for-Android>, 17.05.2011.
- [BI04] Ling Bao und Stephen Intille. Activity Recognition from User-Annotated Acceleration Data. In *Pervasive Computing*, Lecture Notes in Computer Science. Springer, Heidelberg, 2004.
- [BP09] Arno Becker und Marcus Pant. *Android, Grundlagen und Programmierung*. dpunkt, Heidelberg, 1. Auflage, 2009.
- [BVU09] Gerald Bieber, Joerg Voskamp und Bodo Urban. Activity Recognition for Everyday Life on Mobile Phones. In *Universal Access in Human-Computer Interaction. Intelligent and Ubiquitous Interaction Environments*, Lecture Notes in Computer Science. Springer, Heidelberg, 2009.
- [KWM10] Jennifer Kwapisz, Gary Weiss und Samuel Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explorations*, 12, 2010.
- [May06] Horst Mayer. *Beschreibende Statistik*. Hanser, München, 4. Auflage, 2006.
- [NDBB11] Claudia Nickel, Mohammad Derawi, Patrick Bours und Christoph Busch. Scenario Test of Accelerometer-Based Biometric Gait Recognition. *International Workshop on Security and Communication Networks (IWSCN)*, 3, 2011.
- [wek11] <http://www.cs.waikato.ac.nz/ml/weka/>, 17.05.2011.
- [WF00] Ian Witten und Eibe Frank. *Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufman Publishers, San Francisco, 4. Auflage, 2000.