Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projektbericht

Andreas Kamenz

Emotionserkennung mittels Bio-Sensoren - Zeitreihenanalyse

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Andreas Kamenz

Emotionserkennung mittels Bio-Sensoren - Zeitreihenanalyse

Projektbericht eingereicht im Rahmen des Hauptprojektes

im Studiengang Master of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Kai von Luck

Eingereicht am: 25. September 2017

Inhaltsverzeichnis

1. Einleitung

1.1. Motivation

Intelligente Systeme, auch Companion-Systeme genannt, gewinnen in unserem Lebensumfeld immer mehr an Bedeutung. Sie sollen Menschen bei der Bewältigung des Alltags unterstützen, so zum Beispiel als Sprachassistenten wie Apple Siri[1], Google Assistant[2] und Microsoft Cortana[3] oder als Roboter wie Pepper[4] und Buddy[5].

Damit diese Systeme in einer angemessenen Weise auf Personen reagieren können, ist es von Vorteil, den emotionalen Zustand einer Person zu ermitteln. Jedoch ist das Erkennen von Emotionen für Companion-Systeme eine schwierige Aufgabe. In der Forschung gibt es verschiedene Lösungsansätze zur Emotionserkennung basierend auf Audio, Video oder physiologischen Daten. Zusätzlich können mittels multimodaler Verfahren die genannten Datenquellen kombiniert werden. Audio- und videobasierte Verfahren sind weit mehr erforscht als Verfahren, die auf physiologische Daten basieren, sind allerdings prinzipbedingt eingeschränkt auf sprachliche Äußerungen bzw. Gesichtsregungen.

Die physiologischen Daten eines Menschen (Herzfrequenz, Blutdruck, elektrische Leitfähigkeit der Haus, Körpertemperatur, etc.) haben für die Emotionserkennung einige Vorteile. Einerseits können sie nur schwer vorgetäuscht werden. Andererseits unterscheiden sie sich nicht zwischen den verschiedenen menschlichen Kulturkreisen, sondern nur von Individuum zu Individuum.

Die Einsatzgebiete solcher intelligenter technischer Systeme sind vielfältig. Ihr Zweck ist letztendlich, eine individuell auf die Person abgestimmte Reaktion zu ermöglichen bzw. passende

[1] https://www.apple.com/de/ios/siri/
[2] https://assistant.google.com/
[3] https://support.microsoft.com/de-de/help/17214/windows-10-what-is
[4] https://www.ald.softbankrobotics.com/en/robots/pepper
[5] http://www.bluefrogrobotics.com/en/buddy/

Handlungsempfehlungen zu unterbreiten, egal ob bei Computerspielen, im Smart Home, bei Fitness- und Gesundheitsanwendungen oder als Assistenz bei der Planung des Alltags.

1.2. Zielsetzung Hauptprojekt

Das Hauptprojekt soll dazu dienen, die erlangten Erkenntnisse aus dem Grundprojekt und aus den Grund- und Hauptseminaren zu vertiefen und die anhand der im Rahmen des Hauptprojektes durchgeführten Versuchsreihen gewonnenen Daten erste Auswertungen durchzuführen. Zusätzlich werden Recherchen zu den Möglichkeiten zur automatisierten Auswertung der Daten durchgeführt, insbesondere werden analytische Verfahren gesucht, die eine automatisierte Mustererkennung bzw. Klassifikation für das diskrete Emotionsmodell mit 6 Basisemotionen (Wut, Ekel, Furcht, Freude, Traurigkeit, Überraschung) nach Ekman [EFE1972] ermöglichen.

1.3. Aufbau der Arbeit

Diese Ausarbeitung zeigt den aktuellen Stand der Projektarbeit und die noch zu erfüllenden Aufgaben im Hinblick auf die Masterarbeit. Sie ist in drei Kapitel unterteilt. Das erste Kapitel behandelt die Motivation und die Zielsetzung des Projektes. Im zweiten Kapitel wird der Versuchsaufbau erläutert und die bisherigen Auswertungen aufgezeigt. Nachfolgend wird im dritten Kapitel auf mögliche Lösungsansätze für die Zeitreihenanalyse zur automatisierten Emotionserkennung anhand der aus den Versuchen gewonnenen Zeitreihendaten eingegangen. Abschließend wird ein Fazit gezogen und ein Ausblick auf die weiteren Ziele gewährt.

2. Projekt

2.1. EmotionBike

Die in dieser Ausarbeitung aufgezeigten Experimente werden im Rahmen des EmotionBike-Projektes an der HAW Hamburg durchgeführt. Das EmotionBike ist eine Laborumgebung mit dem Ziel, die Interaktion von Menschen mit Computern im Kontext von Exergames zu untersuchen. Die beim Spielen gesammelten Daten können zur Emotionserkennung genutzt werden. Für weitergehende allgemeine Ausführungen wird auf das Grundprojekt [KAM2016] und auf die Veröffentlichungen [MÜL+2015] sowie [MÜL+2016] verwiesen, ein Komponentendiagramm ist in Anhang A dargestellt.

Das EmotionBike System nutzt zur Kommunikation zwischen den einzelnen Knoten eine Publish-Subscribe Architektur, implementiert durch ActiveMQ. Dazu existieren Topics für Steuerbefehle für die Knoten (EMOBIKE.CONTROL) und Statusmeldungen von den Knoten (EMOBIKE.STATUS).

Über das ActiveMQ-Topic namens *EMOBIKE.CONTROL* werden vom Control Center folgende Steuerungsbefehle an die Knoten übermittelt:

sessionInfo	Allgemeine Informationen zum aktuellen Versuchsdurchgang (Proband-ID, Session-ID, Measure-ID, Speicherpfad für die Messdaten)
initializeMeasure	Messung vorbereiten (Verbindung mit Plux-Hub herstellen)
startMeasure	Messung starten
stopMeasure	Messung stoppen
doReset	Alle aktuell laufenden Prozesse kontrolliert abbrechen und auf initialen Zustand zurückkehren
getTime	Aktuellen Zeitstempel des Knoten schicken

Tabelle 2.1.: Steuerungsbefehle

Das zweite Topic *EMOBIKE.STATUS* wird für Statusmeldungen von den Knoten verwendet. Folgende Status sind dabei möglich:

online	Knoten ist erreichbar
ready	Knoten ist bereit für Steuerungsbefehle
measurePrepared	Knoten ist bereit zur Messung
working	Aktive Messung
writing	Daten werden gespeichert
finished	Messung beendet und Daten gespeichert
error	Ein Laufzeitfehler ist aufgetreten
reset	Reset wird durchgeführt

Tabelle 2.2.: Statusmeldungen

Zusätzlich gibt es ein Topic *EMOBIKE.HEARTBEAT*, an dem alle aktiven Knoten periodisch im Sekundentakt eine Meldung schicken, die den aktuellen Zeitstempel des Knotens beinhaltet. Dies dient zum einen zur Erkennung, ob Knoten ohne Fehlermeldung ausgefallen sind. Zum anderen kann im Fall von systemrelevanten Knoten ein Neustart des Versuches erfolgen, um die Datenkonsistenz für den Versuchsdurchlauf zu wahren.

Des Weiteren dient der Heartbeat zur Erkennung von Zeitasynchronitäten zwischen den verschiedenen Knoten, was in einem verteilten System aufgrund von Netzwerklatenzen und unterschiedlicher Zeitermittlungsverfahren zwischen Betriebssystem und Programmiersprache ein nicht zu unterschätzendes Problem darstellt.

Die Notwendigkeit solcher Maßnahmen zur Zeitsynchronisierung hat sich bereits bei den Auswertungen der ersten Probemessungen ergeben. Durch die Verwendung unterschiedlicher Präzisionen (Nanosekunden oder Mikrosekunden) und nicht festgelegter Zeitzone (UTC, MEZ oder MESZ) kam es zu Verwirrungen bezüglich der Datenqualität. Im späteren Verlauf der regulären Versuche gab es noch Abweichungen im Millisekunden- bis einstelligen Sekundenbereich, welche aber mittels Heartbeat-Nachrichten synchronisiert wurden.

2.2. Implementierung des Sensor Service-Knoten

Zum Zweck der Emotionserkennung mithilfe von physiologischen Daten wurden die im Grundprojekt ausgewählten Biosensoren (EDA, EKG, BVP, Respiration und Temperatur) mit dem Plux-Hub[1] verbunden und in die EmotionBike-Umgebung integriert. Dazu wurde eine

[1] http://biosignalsplux.com/en/

Software bestehend aus mehreren Komponenten in Python geschrieben, siehe Anhang B. Dieses Programm wurde als Knoten in das verteilte System des gesamten Versuchsaufbaus integriert und dient zur Steuerung und Datenerfassung vom Plux-Hub.

Die entwickelte Software für die physiologischen Sensoren setzt sich aus insgesamt 5 Komponenten zusammen: Sensor, Plotter, Processing, Communication und Persistence.

Die **Sensor-Komponente** dient als Schnittstelle zum Plux-Hub. Sie besteht aus einer DataReceiver-Klasse, die verschiedene Callback-Methoden implementiert. Diese Callbacks werden vom Plux-Hub aufgerufen, wenn bestimmte Ereignisse eintreten. Dazu zählt die onRawFrame-Methode, in der periodisch die erfassten Sensordaten inklusive Zeitstempel und Sequenznummer in eine Queue zur Weiterverarbeitung eingefügt werden. Die zweite Klasse in der Sensor-Komponente ist der SensorService, der zur Steuerung des Plux-Hubs dient. So baut der Service die Bluetooth-Verbindung auf, sorgt für eine Fehlerbehandlung bei Verbindungsproblemen und startet bzw. beendet eine Messung auf dem Plux-Hub.

Um eine visuelle Kontrolle während der Versuchsdurchführung zu ermöglichen, wurde eine **Plotter-Komponente** entwickelt. Sie stellt in Echtzeit die empfangenen Sensordaten als Diagramme dar (Beispiel siehe Abb. 2.1) und fügt optional die ausgelösten Events als senkrechte Markierungen ein.

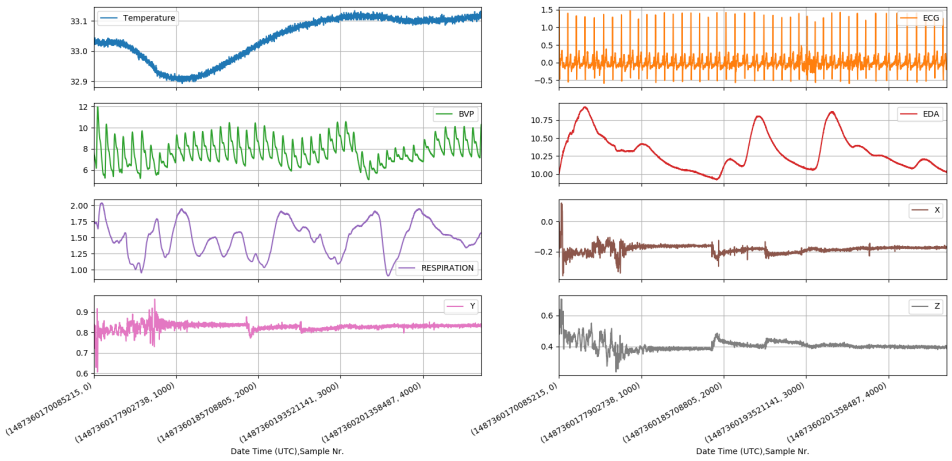


Abbildung 2.1.: Sensordiagramme

Zusätzlich kann die **Processing-Komponente** genutzt werden, um bestimmte Datenanalysen auf ein zuvor konfiguriertes Zeitfenster auszuführen und die Ergebnisse im Diagramm

darzustellen. Beispielsweise können im Diagramm die Peaks markiert oder ein gleitender Durchschnitt eingeblendet werden.

Hierbei mussten jedoch Einschränkungen für die Algorithmen vorgegeben werden, um die Echtzeitfähigkeit zu gewährleisten. So wurden bisher nur Algorithmen mit geringer Komplexität implementiert. Außerdem wurde nicht jedes Datum im Datenstrom verwendet, sondern nur eine Untermenge. Anstatt mit der originalen Abtastrate von 256Hz wurde mit 32Hz bei der Analyse gearbeitet.

Zusätzlich wurde eine Funktion erstellt, die im Nachgang die aufgenommenen Daten verarbeitet und je nach Parametrisierung und verwendeter Algorithmen bestimmte Eigenschaften in den Daten erkennt und in Diagrammen ausgeben kann. Dazu zählen komplexere Zeitreihenanalysen oder solche, die die komplette Sequenz als Eingabe benötigen, wie beispielsweise das globale Maxima, die Baseline oder die Schiefe (engl. *Skewness*).

Der ActiveMQListener aus der **Communication-Komponente** implementiert das STOMP-Protokoll und dient als ActiveMQ-Client. Dieser kann mit dem ActiveMQ-Broker, der die Topic-Nachrichten für sämtliche ActiveMQ-Clients vorhält, kommunizieren. Weitere Clients sind beispielsweise die Thermo-Kamera oder die Kinect-Kamera.

Außerdem repräsentiert der ActiveMQListener einen Zustandsautomaten mit 8 Zuständen, der nur mit den richtigen Steuerungsbefehlen in den nächsten Zustand übergeht (siehe Abb. 2.2). Falls es zu Störungen oder Problemen kommt, dann wird in den Fehlerzustand gewechselt und eine Fehlermeldung an das Topic EMOBIKE.STATUS gesendet. Dies dient dazu, dass das Control Center zu jeder Zeit weiß, in welchem Zustand sich alle Knoten befinden und je nach Bedeutsamkeit eines ausgefallenen Knotens ein Reset des gesamten Versuchsaufbaus einleiten kann.

Die zweite Klasse der Communication-Komponente ist der CommunicationService. Dieser stellt sicher, dass die Kommunikationskanäle (Queues) zwischen den Komponenten aufgebaut sind, und überwacht deren Status, um bei Problemen in den Fehlerzustand zu wechseln.

Die **Persistence-Komponente** bekommt über eine Data-Queue die einzelnen Messwerte der Sensoren zugeschickt und speichert diese in einer CSV-Datei in dem aus der SessionInfo bekannten Pfad ab. Auf Grund der hohen Frequenz der eingehenden Messwerte werden die Daten temporär zwischengespeichert und erst am Ende einer Messung einmalig auf den Datenträger abgespeichert.

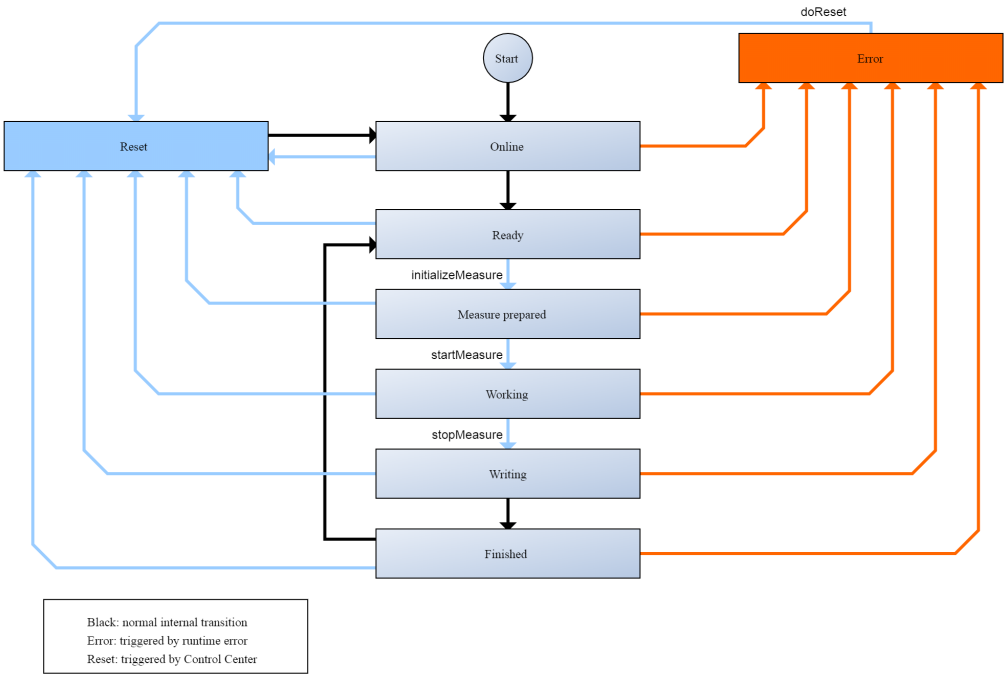


Abbildung 2.2.: Zustandsautomat

Insgesamt werden 8 verschiedene Threads gestartet, die über Queues miteinander kommunizieren. MainThread, AMQListener, CommunicationService, SensorService, DataReceiver, Graph (Plotter), Utilities (Processing) und CSVWriter sind dabei jeweils einzelne Threads. Dies war erforderlich, um die parallele Ausführung verschiedener Aufgaben zu ermöglichen und eine permanente Erreichbarkeit für Steuerungssignale vom Control Center, bzw. Daten- und Statusnachrichten vom Plux-Hub sicherzustellen (siehe Anhang B).

2.3. Ergebnisse der Datenauswertung

In den Beiträgen ”Enhancing Exercise Experience with Individual Multi-Emotion Provoking Game Elements” bei IEEE Symposium Series on Computational Intelligence (IEEE SSCI 2017) und ”Emotional Journey for an Emotion Provoking Cycling Exergame” bei IEEE 4th Intl. Conference on Soft Computing & Machine Intelligence (ISCMI 2017) werden erste Ergebnisse der Versuchsreihen erläutert.

Dort wird exemplarisch anhand der EDA-Daten aufgezeigt, dass durch die Kombination von visueller Emotionserkennung und physiologischer Daten die Erkennungsrate verbessert werden kann und zusätzlich eine robustere Klassifizierung möglich wird.

Beispielsweise betrug die Erkennungsrate in einem Level, welches Angst und Überraschung auslösen soll, mittels visueller Emotionserkennung 54,2%. Die EDA-Daten zeigten zusätzlich bei 83,3% aller Events einen starken Anstieg. Auf Grund dieser Erkenntnis konnte letztendlich durch die Kombination beider Datenquellen eine Erkennungsrate von 87,5% für dieses Level erreicht werden.

3. Lösungsansätze zur Zeitreihenanalyse

Im Folgenden wird der aktuelle Forschungsstand im Bereich Zeitreihenanalyse aufgezeigt. Dabei wird das Ziel verfolgt, mehrere aktuelle Lösungsansätze zu präsentieren, die in der Lage sind, zeitbehaftete Sensordaten zu verarbeiten und die entsprechenden Emotionen zu erkennen.

Um die Sensordaten zu analysieren, muss zunächst ein passendes mathematisches Modell gefunden werden, das die wichtigsten Aspekte der im vorherigen Kapitel beschriebenen Problemstellung widerspiegelt.

Typischerweise werden die von einem Sensor gelieferten Daten als eine Zeitreihe modelliert. Das Messsignal des Sensors wird innerhalb eines endlichen Zeitintervalls in regelmäßigen Abständen aufgezeichnet, sodass nach l Beobachtungen eine Folge von Messwerten entsteht:

$$x_1 \to x_2 \to \cdots \to x_l, \text{ bzw. } (x_1, \ldots, x_l)$$

Im Fall von m Sensoren liegen m unterschiedliche Zeitreihen vor, die unabhängig voneinander betrachtet werden müssen:

$$S_{i,j} = (x_{i,j}(t_1), x_{i,j}(t_2), \ldots, x_{i,j}(t_{l(j)})),$$

wobei $x_{i,j}$ der Messwert von Sensor j ($j \in 1 \ldots m$) im Zeitpunkt t für den Probanden i mit $i \in 1 \ldots n$ ist. Die Länge jeder Zeitreihe kann bei Sensoren unterschiedlich sein ($l = l(j)$). Insgesamt ergeben sich also $n \cdot m \cdot \sum_{j=1}^{m} l(j)$ Messwerte. Darüber hinaus ist ein Zielvektor vorgegeben:

$$Y = (y_1, \ldots, y_n),$$

wobei y_i den Zustand des Probanden i, also konkret seine Emotion beschreibt.

Das Ziel der Sensordatenanalyse ist es, bestimmte Muster (engl. *Features*) im Zeitreihenverlauf zu erkennen und diese Muster den verwendeten Emotionen zuzuordnen. Somit soll zukünftig

für beliebige Daten eines Sensors eine Aussage ermöglicht werden, um welche Emotion es sich bei diesen Daten handelt.

Die Lösungsverfahren für die oben formulierte Problemstellung werden dem Forschungsbereich der **Klassifikation von Zeitreihen** (engl. *time series classification*, kurz TSC) zugeordnet. Diese Verfahren können in folgende Kategorien unterteilt werden:

- Formbasierte Klassifikation (engl. *shape based classification*)
- Feature-basierte Klassifikation (engl. *feature based classification*)
- Neuronale Netze basierte Klassifikation (engl. *neural network based classification*)

Jede dieser Kategorien versucht auf eigene Art und Weise, die Zeitreihendaten zu klassifizieren. Im folgenden werden die Kategorien im einzelnen beschrieben und die Unterschiede aufgezeigt.

3.1. Formbasierte Klassifikation

Formbasierte Verfahren beschäftigen sich lediglich mit der Form der Zeitreihenkurve und ermöglichen eine Mustererkennung, indem sie auf bestimmte Weise die Distanz zwischen zwei Kurven berechnen. Dabei sind bestimmte Zeitreihen vorgegeben, deren Zugehörigkeit zu den gegebenen Klassen bekannt ist. Anschließend wird mithilfe eines k-Nearest-Neighbor-Algorithmus ermittelt, zu welchen bekannten Zeitreihen eine neue Zeitreihe am nächsten ist, und somit ihre Klassenzugehörigkeit bestimmt. Das bekannteste und erfolgreichste von den formbasierten Verfahren ist ”Dynamic Time Warping” (DTW) [MÜL2007].

3.1.1. Dynamic Time Warping

Beim DTW werden zwei zeitabhängige wertnormierte Signalfolgen verglichen, siehe Abb. 3.1:

$$X := (x_1, x_2, ..., x_n) \text{ und } Y := (y_1, y_2, ..., y_m)$$

Jedes x_i wird auf ein y_j abbildet, wobei mehrere x auf ein y, bzw. mehrere y auf ein x zugeordnet sein können. Das Verfahren stammt ursprünglich aus der automatischen Spracherkennung und soll dabei Variationen durch unterschiedlich schnell gesprochene Wörter erkennen, es kann aber auch für andere zeitabhängige Daten verwendet werden.

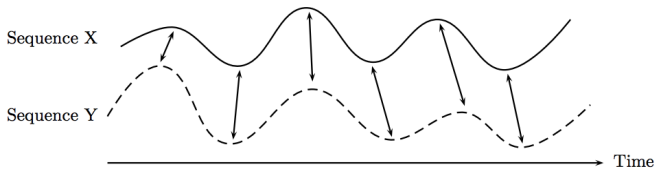


Abbildung 3.1.: Signalfolgen X und Y [MÜL2007]

Mittels einer Kostenfunktion $c(x_i, y_j)$ werden nah beieinander liegende Punkte mit geringen Kosten versehen, weit auseinanderliegende Punkte dagegen mit hohen Kosten. Es wird nachfolgend eine Kostenmatrix M für das Kreuzprodukt aller Elemente aus X und Y erstellt. Eine grafische Darstellung der Kostenmatrix ist in Abb. 3.2 ersichtlich. Dabei sind helle Stellen im Diagramm hohe Kosten und dunkle Stellen - geringe Kosten. Ziel des Algorithmus ist es, anhand dieser Kostenmatrix den optimalen, günstigsten Pfad (sog. "Warping Path") zu finden. Die Pfadsuche beginnt bei $c(x_1, y_1)$ und endet bei $c(x_n, y_m)$. Als Einschränkungen bei der Pfadsuche gelten ein monotoner Anstieg und eine festgelegte maximale Schrittweite von 1.

Der resultierende Pfad ist somit eine Liste von einzelnen Kosten, die summiert die Gesamtkosten ergeben. Mit Hilfe dieser Gesamtkosten kann nun festgestellt werden, wie ähnlich sich zwei Signalfolgen sind. Im Prinzip führt der Algorithmus somit eine adaptive Zeitnormierung durch. Das heißt, es findet eine Streckung bzw. Stauchung der Signalfolge statt.

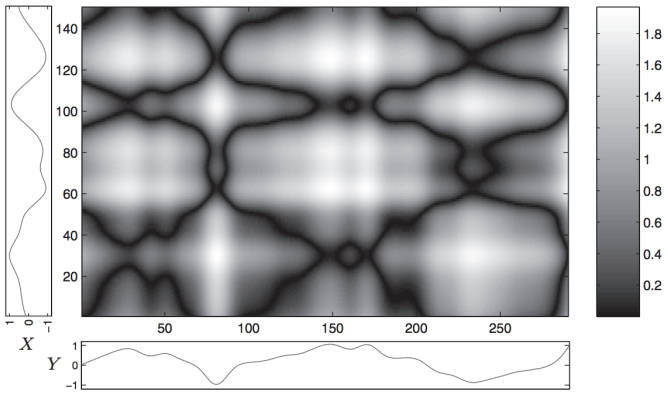


Abbildung 3.2.: Kostenmatrix für $X \times Y$ [MÜL2007]

DTW liefert laut [CKF2016] die besten Ergebnisse für eine Zeitreihenklassifikation auf vielen Benchmark-Datasets. Es ist allerdings sehr speicherintensiv und skaliert schlecht mit wachsender Zeitreihenlänge [BAG+2017].

3.2. Feature-basierte Klassifikation

Feature-basierte Verfahren extrahieren aus jeder Zeitreihe eine große Anzahl an generischer Features, mithilfe dessen die Zeitreihenklassifikation erfolgt. Dabei wird unterschieden nach [NAM2001]:

- Features erster Ordnung - direkt aus der vorliegenden Zeitreihe berechnet.
- Features zweiter Ordnung - ursprüngliche Zeitreihe muss zunächst transformiert werden, z.B. folgendermaßen: $f(t) = f(t + D) - f(t)$. Wobei f die Wertefunktion der Zeitreihe ist, t der Zeitpunkt und D ein benutzerdefinierter Wert. Das Resultat sind Features, die auf die Differenz der Zeitpunkte basieren und somit lokale Abhängigkeiten abbilden.

Nach [CKF2016] gibt es ca. 1000 verschiedene mögliche Features für die Zeitreihenanalyse, wovon meist ca. 50 verwendet werden. Beispiele dafür sind u.a. das artihmentische Mittel, die Standardabweichung, die Schiefe (Skewness), die Wölbung, der Median, die Anzahl der Peaks mit einem bestimmten Anstieg, die Periodizität der Peaks, der globale Trend, usw.

Die Feature-Extraktion dient der Dimensionsreduktion, da jede Zeitreihe durch eine Menge von statistisch signifikanten Features repräsentiert werden kann. Anschließend können die Feature-Vektoren mithilfe von Klassifikationsalgorithmen wie AdaBoost oder Random Forest klassifiziert werden. Es muss somit nicht mehr jeder einzelne Zeitpunkt der Zeitreihe für die Klassifikation betrachtet werden, sondern nur die berechneten Features.

AdaBoost ist ein von Freund und Schapire [FS1995] veröffentlichtes Boosting-Verfahren, dass viele schwache Klassifikatoren zu einem starken Klassifikator kombiniert. Das Random Forest Verfahren [BRE2001] basiert dagegen auf viele voneinander unabhängige und zufällig erzeugte Entscheidungsbäume. Die Entscheidungen aus allen Entscheidungsbäumen werden bei der Klassifikation gesammelt und die am häufigsten gewählte Klasse wird als Endergebnis zurückgeliefert.

Feature-basierte Klassifikationsverfahren haben den Vorteil, dass sie bedeutend schneller sind als formbasierte Verfahren, da nicht mehr jeder einzelne Datenpunkt der Zeitreihe für die Klassifikation betrachtet werden muss, sondern nur die berechneten Features. Der große Nachteil ist jedoch, dass es sehr viel Aufwand bedeutet, die richtigen Features für eine bestimmte Problemstellung auszuwählen und eventuelle Datenvorverarbeitungen erforderlich sind.

3.3. Neuronale Netze basierte Klassifikation

Direkte Verfahren wie neuronale Netze oder andere Deep Learning Modelle lernen die Features aus den Zeitreihen automatisch. Durch die stetig ansteigende Menge an Rechenkapazität werden Neuronale Netze basierte Klassifikationsverfahren immer leistungsfähiger und zählen mittlerweile vor allem bei sehr großen Datenmengen zu den bevorzugten Verfahren. Typischerweise werden für Zeitreihen Netzwerkklassen wie ”Long Short-Term Memory” (LSTM) oder ”Convolutional Neural Networks” (CNN) eingesetzt.

LSTM Netze wurden erstmals 1997 von Hochreiter und Schmidhuber beschrieben [HS1997] und basieren auf Recurrent Neural Networks, die insbesondere in der Lage sind, zeitliche Zusammenhänge in Datenreihen zu erkennen.

Das klassische Einsatzgebiet von CNNs ist hingegen die Objektklassifizierung durch Feature-Extraktion anhand von 2D- oder 3D-Bildern. In diesem Bereich sind sie zum Standard geworden und liefern state-of-the-art Erkennungsraten. Da Zeitreihen jedoch im Gegensatz zu Bildern eindimensional sind, müssten die CNNs auf 1D-Daten angepasst werden. Dazu existieren bereits erste erfolgreiche Versuche, siehe [ZHA+2017] und [WYO2017].

Der große Vorteil der Deep Learning Modelle ist, dass die Features nicht mehr generisch oder manuell festgelegt sind, sondern sie werden für jede spezifische Problemstellung vom Netz passend erlernt. D.h. diese Modelle können die Problemstellung präziser beschreiben als die beiden anderen Kategorien von Zeitreihenklassifikationsverfahren. Sie erfordern allerdings meistens sehr viele Trainingsdaten, was in der Praxis häufig eine große Herausforderung darstellt. Bei zu wenigen Trainingsdaten sind viele Anpassungen bzw. Optimierungen nötig, um gute Ergebnisse zu erhalten.

4. Zusammenfassung

4.1. Ergebnisse

Im Rahmen des Hauptprojektes wurde ein funktionstüchtiges System zur Gewinnung der physiologischen Daten der Probanden im Versuchsaufbau konzipiert und umgesetzt. Dabei wurde das Plux Toolset mit den Sensoren für EDA, ECG, BVP, Respiration und Temperatur in die Publish-Subscribe Architektur des EmotionBike Projektes integriert.

Bei den anschließenden Versuchsreihen mit 25 Probanden wurden die entsprechenden Sensordaten aufgezeichnet, aufbereitet und daraus ein Dataset erstellt.

Zusätzlich wurde nach Lösungsverfahren gesucht, die zur automatisierten Emotionserkennung in Frage kommen können. Diese versuchen, anhand der zeitbehafteten Daten bestimmte (generische) Features zu extrahieren oder anhand der Datenkurve zu klassifizieren.

Somit wurden die Grundlagen für die nachfolgende Implementierung der automatisierten Emotionserkennung gelegt.

4.2. Probleme und Risiken

Im Zuge einer automatischen Emotionserkennung beim EmotionBike stellt sich die Frage, ob die Klassifikationsalgorithmen für diesen Anwendungsfall (Emotionserkennung) zu komplex sind und dadurch nur eine nachträgliche Analyse der Daten möglich ist oder ob auch eine Echtzeiterkennung erreichbar ist. Außerdem muss die Qualität der Sensordaten überprüft und ausgewertet werden. Hierbei kann es passieren, dass das ohnehin kleine Dataset noch weiter verkleinert werden muss, wenn es in einzelnen Durchgängen zu fehlerhaften Messungen gekommen ist.

Beim Einsatz von den auf neuronalen Netzen basierten Klassifizierungsverfahren werden die in den Versuchsreihen gewonnenen Daten vermutlich nicht ausreichen, um eine gute Generalisie-

rung des Modells zu erreichen. Hier wird eventuell eine Suche nach neuen Datasets nötig oder es müssen aus den bestehenden Daten weitere synthetische Daten generiert werden.

Des Weiteren ist der Aufwand für die Fusion mit den videobasierten Daten noch nicht abschätzbar, weshalb sie ein optionales Unterthema der Masterarbeit ist.

4.3. Weitere Ziele

Auf Basis der im Hauptprojekt gewonnenen Daten und untersuchten Lösungsverfahren ist für die Masterarbeit eine Untersuchung der folgenden Fragen geplant:

1. Ob und wie gut ist eine Emotionserkennung nur auf Basis von physiologischen Daten möglich?
2. Liefert eine Fusion mit anderen Daten wie Video immer eine bessere Erkennungsrate?

Literatur

Literatur

[EFE1972] Ekman, P. u. a. *Emotion in the human face: guide-lines for research and an integration of findings*. Pergamon general psychology series. Pergamon Press, 1972. URL: https://books.google.de/books?id=MeB9AAAAMAAJ.

[MÜL2007] Müller, Meinard. *Information Retrieval for Music and Motion*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007. ISBN: 3540740473.

Artikel

[BAG+2017] Bagnall, Anthony u. a. "The Great Time Series Classification Bake off: A Review and Experimental Evaluation of Recent Algorithmic Advances". In: *Data Min. Knowl. Discov.* 31.3 (Mai 2017), S. 606–660. ISSN: 1384-5810. DOI: 10.1007/s10618-016-0483-9.

[BRE2001] Breiman, Leo. "Random Forests". In: *Mach. Learn.* 45.1 (Okt. 2001), S. 5–32. ISSN: 0885-6125. URL: https://doi.org/10.1023/A:1010933404324.

[CKF2016] Christ, Maximilian u. a. "Distributed and parallel time series feature extraction for industrial big data applications". In: (Okt. 2016).

[FS1995] Freund, Yoav und Robert E. Schapire. "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting". In: EuroCOLT '95 (1995), S. 23–37. URL: http://dl.acm.org/citation.cfm?id=646943.712093.

[HS1997] Hochreiter, Sepp und Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), S. 1735–1780. ISSN: 0899-7667. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[KAM2016] Kamenz, Andreas. "Emotionserkennung mittels Bio-Sensoren". In: *Projektberichte HAW* (2016). URL: http://users.%20informatik.haw-hamburg.de/~ubicomp/projekte/master2016-proj/kamenz.pdf.

[MÜL+2015] Müller, Larissa u. a. "Entertainment Computing - ICEC 2015: 14th International Conference, ICEC 2015, Trondheim, Norway, September 29 - Ocotober 2, 2015, Proceedings". In: (2015). Hrsg. von Konstantinos Chorianopoulos u. a., S. 155–168. URL: http://dx.doi.org/10.1007/978-3-319-24589-8_12.

[MÜL+2016] Müller, Larissa u. a. "Physiological data analysis for an emotional provoking exergame". In: (Dez. 2016), S. 1–8. DOI: 10.1109/SSCI.2016.7850042.

[NAM2001] Nanopoulos, Alex u. a. "Feature-based Classification of Time-series Data". In: 10 (Jan. 2001), S. 49–61.

[WYO2017] Wang, Z. u. a. "Time series classification from scratch with deep neural networks: A strong baseline". In: (Mai 2017), S. 1578–1585. DOI: 10.1109/IJCNN.2017.7966039.

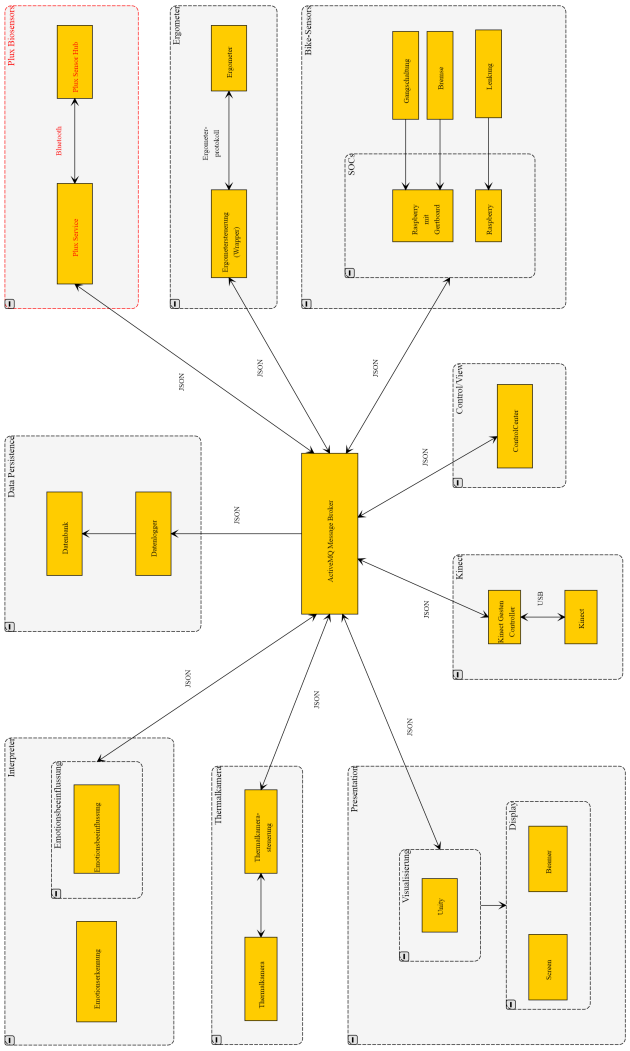
[ZHA+2017] Zhao, B. u. a. "Convolutional neural networks for time series classification". In: *Journal of Systems Engineering and Electronics* 28.1 (Feb. 2017), S. 162–169. DOI: 10.21629/JSEE.2017.01.18.

Tabellenverzeichnis

Abbildungsverzeichnis

Anhänge

A. EmotionBike Komponentendiagramm



B. Sensor Service Komponentendiagramm