



Durchgeführt im Rahmen des
BMBF-Projekt: ELISE

Dokumentation der Projektarbeit

Entwurf eines kompakten mikrocontrollergestützten Systems zur
Emotionserkennung in einer Virtual-Reality-Umgebung

WiSe 2017/2018 und SoSe 2018

Projektbetreuer:

Medizinische Informatik und Mikrosystementwurf

Prof. Dr. rer. nat. Rainer Brück

Dr.-Ing. Armin Grünewald

M.Sc. David Krönert

M.Sc. Tanja Eiler

Forschungsgruppe für Mustererkennung

Prof. Dr.-Ing. Marcin Grzegorzek

M.Sc. Frédéric Li

Projektteilnehmer:

Artur Piet (Sprecher der Projektgruppe)

Jonas Pöhler (Stellv. Sprecher der Projektgruppe)

Arnaud Eric Toham Waffo

Boris Kamdem

Kevin Orth

Meryem Dural

Minas Michail

Inhaltsverzeichnis

1	Einleitung	8
1.1	Hintergrund und Motivation	8
1.2	ELISE Projektbeschreibung	8
1.3	Gliederung dieser Dokumentation	8
1.4	Anhang	8
2	Organisation	9
2.1	Verantwortungsbereiche	10
2.2	Gruppentreffen	10
3	Grundlagen	11
3.1	Definition von Emotionen	11
3.2	Virtual Reality (VR)	11
3.3	Sensoren und biophysiological Signale zur Emotionserkennung . .	11
3.3.1	Körpertemperatur-Sensor	11
3.3.2	Blood Volume Pulse-Sensor (BVP)	11
3.3.3	Messen der Sauerstoffsättigung (SpO2)	11
3.3.4	Galvanic Skin Response (GSR)	11
3.3.5	Elektroenzephalografie (EEG)	11
3.3.6	Elektrookulografie (EOG)	11
3.3.7	Analog/Digital-Wandler	11
3.4	Kommunikation	11
3.5	Grundlagen der Mustererkennung	11
3.6	Emotion Recognition Chain	13
3.6.1	Datenerfassung	13
3.6.2	Vorverarbeitung	13
3.6.3	Segmentation	14
3.6.4	Merkmalsextraktion	14
3.6.5	Klassifikation	14
4	State-of-the-Art Analyse	19
I	Erster Prototype	20
<hr/>		
5	Systementwurf und Konzept	20
5.1	Anforderungen	20
5.2	Konzept	20

5.3	Hardwareauswahl	20
5.3.1	Auswahlkriterien	20
5.3.2	Festlegung der genutzten Hardware	20
5.4	Hardwarearchitektur	20
5.4.1	GSR-Sensor	20
5.4.2	Temperatur-Senosr	20
5.4.3	Pulsoximeter	20
5.4.4	EEG	20
5.4.5	EOG	20
5.4.6	Datenübertragung	20
5.5	Programmierung	20
5.6	Aufnahme der übertragenen Daten	20
6	Realisierung	21
7	Emotionsinduktion	22
7.1	Ablauf	22
7.2	Fragebogen	22
7.3	Szenarien	22
7.3.1	Glück	22
7.3.2	Langeweile	22
7.3.3	Frustration	22
8	Messreihe	23
9	Mustererkennung	24
9.1	Datenerfassung	24
9.2	Vorverarbeitung	24
9.3	Segmentation	24
9.4	Merkmalsextraktion	25
9.4.1	Handgefertigte Merkmale	25
9.4.2	Codebook Approach	28
9.5	Klassifikation	31
10	Ergebnisse	32
10.1	Ergebnisse der hand-gefertigten Merkmale	32
10.2	Ergebnisse des Codebook Approach	32
10.3	Analyse der Ergebnisse	32
II	Zweiter Prototype	33

11 Systementwurf und Konzept	33
11.1 Anforderungen	33
11.2 Konzept	33
11.3 Hardwareauswahl	33
11.3.1 Auswahlkriterien	33
11.3.2 Festlegung der genutzten Hardware	33
11.4 Hardwarearchitektur	33
11.4.1 GSR-Sensor	33
11.4.2 Temperatur-Sensor	33
11.4.3 Pulsoximeter	33
11.4.4 EEG	33
11.4.5 EOG	33
11.4.6 Datenübertragung	33
11.5 Programmierung	33
11.6 Aufnahme der übertragenen Daten	33
12 Realisierung	34
13 Emotionsinduktion	35
13.1 Ablauf	35
13.2 Fragebogen	35
13.3 Szenarien	35
13.3.1 Glück	35
13.3.2 Langeweile	35
13.3.3 Frustration	35
14 Messreihe	36
15 Mustererkennung	37
15.1 Merkmalsextraktion für Emotionserkennung	37
15.1.1 Hand-gefertigte Merkmale	37
15.1.2 Codebook Approach	37
16 Ergebnisse	38
16.1 Ergebnisse der hand-gefertigten Merkmale	38
16.2 Ergebnisse des Codebook Approach	38
16.3 Analyse der Ergebnisse	38
III Dritter Prototype	39

17 Systementwurf und Konzept	39
17.1 Anforderungen	39
17.2 Konzept	39
17.3 Hardwareauswahl	39
17.3.1 Auswahlkriterien	39
17.3.2 Festlegung der genutzten Hardware	39
17.4 Hardwarearchitektur	39
17.4.1 GSR-Sensor	39
17.4.2 Temperatur-Sensor	39
17.4.3 Pulsoximeter	39
17.4.4 EEG	39
17.4.5 EOG	39
17.4.6 Datenübertragung	39
17.5 Programmierung	39
17.6 Aufnahme der übertragenen Daten	39
18 Realisierung	40
19 Emotionsinduktion	41
19.1 Ablauf	41
19.2 Fragebogen	41
19.3 Szenarien	41
19.3.1 Glück	41
19.3.2 Langeweile	41
19.3.3 Frustration	41
20 Messreihe	42
21 Mustererkennung	43
21.1 Merkmalsextraktion für Emotionserkennung	43
22 Ergebnisse	44
22.1 Ergebnisse der hand-gefertigten Merkmale	44
22.2 Ergebnisse des Codebook Approach	44
22.3 Analyse der Ergebnisse	44
IV Vierter Prototype	45

23 Systementwurf und Konzept	45
23.1 Anforderungen	45
23.2 Konzept	45
23.3 Hardwareauswahl	45
23.3.1 Auswahlkriterien	45
23.3.2 Festlegung der genutzten Hardware	45
23.4 Hardwarearchitektur	45
23.4.1 GSR-Sensor	45
23.4.2 Temperatur-Sensor	45
23.4.3 Pulsoximeter	45
23.4.4 EEG	45
23.4.5 EOG	45
23.4.6 Datenübertragung	45
23.5 Programmierung	45
23.6 Aufnahme der übertragenen Daten	45
24 Realisierung	46
25 Emotionsinduktion	47
25.1 Ablauf	47
25.2 Fragebogen	47
25.3 Szenarien	47
25.3.1 Glück	47
25.3.2 Langeweile	47
25.3.3 Frustration	47
26 Messreihe	48
27 Mustererkennung	49
27.1 Merkmalsextraktion für Emotionserkennung	49
28 Ergebnisse	50
28.1 Ergebnisse der hand-gefertigten Merkmale	50
28.2 Ergebnisse des Codebook Approach	50
28.3 Analyse der Ergebnisse	50
29 Zusammenfassung und Ausblick	51
29.1 Zusammenfassung	51
29.2 Fazit	51
29.3 Ausblick	51
Abbildungsverzeichnis	52

Tabellenverzeichnis	53
Abkürzungen	54
Anhang	55

1 Einleitung

Verantwortlich: Minas

- Next Step: Bitte selbst bei github hochladen oder Artur zuschicken

1.1 Hintergrund und Motivation

1.2 ELISE Projektbeschreibung

1.3 Gliederung dieser Dokumentation

1.4 Anhang

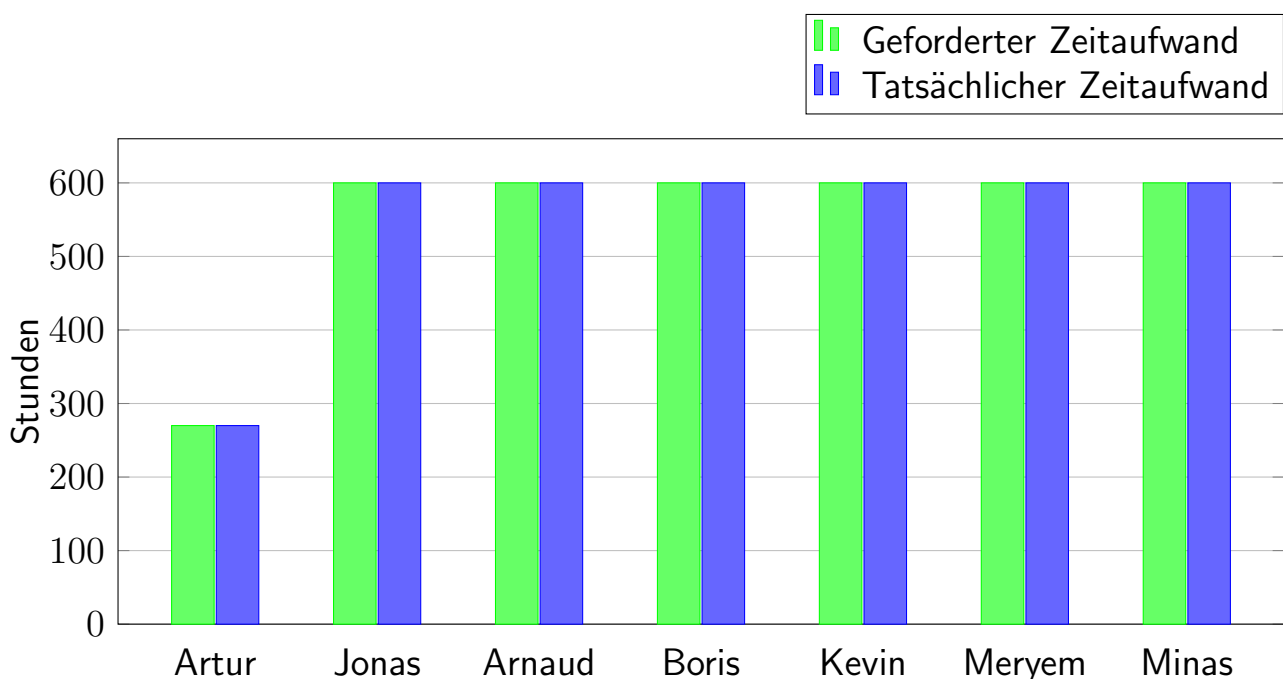
2 Organisation

Verantwortlich: Artur

Diese Projektgruppe des ELISE Projektes wird dem Lehrstuhl Medizinische Informatik & Mikrosystementwurf und dem Lehrstuhl für Mustererkennung zugeordnet.

Die Gruppe wird von Dr.-Ing. Armin Grünewald, David Krönert und Frédéric Li geleitet. Innerhalb der Gruppe wurden dazu unabhängig ein Sprecher und ein stellvertretender Sprecher von den Gruppenmitgliedern gewählt. Als Projektgruppensprecher wurde Artur Piet und als Stellvertreter Jonas Pöhler ausgewählt. Diese Stellung ist jedoch nicht die eines Leiters mit Entscheidungs- und Weisungsbefugnissen. Die Sprecher sind also auf die Kooperationsbereitschaft der anderen Gruppenmitglieder angewiesen. Konkrete Aufgaben der Sprecher waren unter anderem das Verteilen von Verantwortungsbereichen auf alle Mitglieder, die Einführung von regelmäßigen Gruppentreffen um den Überblick über alle Fortschritte zu garantieren und die Übernahme möglichst aller organisatorischer Tätigkeiten (z.B. Messreihen organisieren, Beschaffungsprozesse koordinieren, usw.).

Die Laufzeit der Projektgruppe wurde auf etwa 1 Jahr gesetzt, wobei das Kick-Off Meeting am 23.10.2017 stattfand. Der geforderte individuelle Zeitaufwand aller Gruppenmitglieder entspricht der jeweils im Modulhandbuch des Studienganges definierten Leistungspunkte für die Projektarbeit, also 600 Stunden für Jonas Pöhler, Arnaud Eric Toham Waffo, Boris Kamdem, Kevin Orth, Meryem Dural sowie Minas Michail und 270 Stunden für Artur Piet. Es folgt ein Balkendiagramm mit den geforderten Stunden im Vergleich zum tatsächlichem Zeitaufwand.



2.1 Verantwortungsbereiche

Innerhalb der Projektgruppe (PG) war ein Ziel, dass jeder der Mitglieder "Experte" für einen Bereich wird. Damit haben wir die Verantwortung relativ gleichmäßig auf alle aufgeteilt. Dazu muss noch erwähnt werden, dass die Teammitglieder nicht nur ausschließlich die Aufgaben des jeweiligen Verantwortungsbereiches erledigt habe. Es wurde sich vielmehr gegenseitig immer unterstützt und viel zusammengearbeitet. Die Verantwortungsbereiche haben sich mit der Zeit folgendermaßen aufgeteilt:

- Artur Piet: Mustererkennung, 3D-Konstruktion und Sprecher der PG
- Jonas Pöhler: Hardware, Webanbindung und Stellv. Sprecher der PG
- Arnaud Eric Toham Waffo: Elektrodenauswahl
- Boris Kamdem: Langeweile-Szenario und Fragebogen in VR
- Kevin Orth: Komplette Hardware
- Meryem Dural: Frustrations-Szenario in VR
- Minas Michail: Glücks-Szenario in VR

2.2 Gruppentreffen

Die wöchentliche Gruppentreffen finden immer Donnerstags ab 14:00 Uhr statt und beginnen in der Regel mit einer Update-Runde. Hierbei kommen alle Gruppenteilnehmer der Reihe nach dran und jeder erklärt kurz woran letzte Woche gearbeitet wurde, was die aktuellen Herausforderungen und Probleme sind und was genau als nächstes geplant ist. Ziel ist es den Austausch und die Kommunikation unter den Teammitgliedern und den Projektleitern zu fördern, damit alle Teilnehmer auf dem selben Wissenstand sind, da einzelnen Aufgaben durchaus großen Einfluß auf Tätigkeiten von anderen Mitgliedern haben können.

3 Grundlagen

Verantwortlich: Arnaud

3.1 Definition von Emotionen

Verantwortlich: Arnaud

3.2 Virtual Reality (VR)

Verantwortlich: Arnaud, Boris

3.3 Sensoren und biophysiological Signale zur Emotionserkennung

Verantwortlich: Arnaud, Kevin

3.3.1 Körpertemperatur-Sensor

3.3.2 Blood Volume Pulse-Sensor (BVP)

3.3.3 Messen der Sauerstoffsättigung (SpO2)

3.3.4 Galvanic Skin Response (GSR)

3.3.5 Elektroenzephalografie (EEG)

3.3.6 Elektrokulografie (EOG)

3.3.7 Analog/Digital-Wandler

3.4 Kommunikation

Verantwortlich: Kevin, Jonas

3.5 Grundlagen der Mustererkennung

Verantwortlich: Artur

- Bereit zum Korrekturlesen.

Mustererkennung (enlg. "pattern recognition") ist ein Unterthema des maschinellen Lernens. Das Ziel besteht darin, automatisierte Systeme zu entwerfen, die hoch abstrakte Muster in Daten erkennen können. Konkret heißt dies, dass man Maschinen beibringen möchte komplexer Aufgaben zu lösen, welche vom Menschen nahezu mühelos und natürlich erledigt werden können. Typische Beispiele für die zahlreichen Anwendungsbereiche sind die Objekterkennung, Spracherkennung sowie die

Erkennung und Verfolgung in Bildern. Die Emotionserkennung ist ein Anwendungsbereich der Mustererkennung. Die Hauptidee hinter der Lösung eines Mustererkennungs-Problems ist es, dieses als Klassifikationsproblem zu übersetzen, wobei die zu erkennende Mustern die unterschiedliche Klassen bilden. Die vom Mustererkennungs-System eingegebenen Daten werden dann verarbeitet und der “am nächsten liegenden” Klasse zugeordnet. Beispielsweise können bei der Emotionserkennung die Eingangsdaten Bilder oder physiologische Signale sein, die in verschiedene Klassen eingeteilt werden, welche jeweils einer Emotion entsprechen.

Ein wichtiger Teil eines jeden Mustererkennung-Problems ist der Lernansatz, mit welchem die Maschine lernen soll die Muster in den Daten zu erkennen. Traditionell werden zwei Ansätze verwendet:

- Überwachter Lernansatz: Dieser Ansatz kann nur verwendet werden, wenn vor der Verarbeitung der Daten ein Datenbeschriftungsschritt durchgeführt wurde. In diesem Schritt wird jedem Element des Datensatzes ein Etikett (engl. “label”) zugewiesen, das angibt, welcher Klasse der jeweilige Datenpunkt zugeordnet werden kann. Die zusätzlichen Informationen, die die Etiketten liefern, werden als Grundlage verwendet, um sie mit der Vorhersage des Systems zu vergleichen und zu korrigieren, wenn sie nicht gleich sind.
- Unüberwachter Lernansatz: Dieser Ansatz wird verwendet, wenn keine Etiketten für die Daten vorhanden sind. Unüberwachte Lerntechniken zielen darauf ab, der Maschine beizubringen, Muster in den Daten selbst zu finden. Sie werden meist verwendet, um Einblicke in Daten zu erhalten, deren Struktur unbekannt ist.

Überwachtes Lernen liefert aktuell weit bessere Ergebnisse, jedoch ist die Beschriftung mit Etiketten der Daten nicht immer einfach oder teilweise sogar gar nicht möglich (z.B. wenn die Datenmenge sehr groß ist oder wenn Unsicherheit über die Vergabe der Etiketete besteht). Aus diesem Grund wächst das Interesse an unüberwachten Lernansätzen. Diese Ansätze sind jedoch schwierig zu benutzen, da sie eine große Menge an Daten voraussetzen. Kompromisse sind mit semi-überwachten Lernansätzen möglich, bei denen die Daten für einen Teil des Datensatzes (aber nicht für den ganzen Datensatz) mit Etiketten beschriftet und damit bekannt sind. In diesem Fall kann eine Mischung aus überwachten und Unüberwachten Techniken angewendet werden [1].

Im Rahmen des ELISE-Projekts werden mit Hilfe von Mustererkennungsverfahren eindimensionale Zeitsignale von physiologischen Sensoren in Echtzeit für die Erkennung von drei Emotionen verarbeitet: Glück, Frustration und Langeweile. Um den Emotionsklassifizierer aufzubauen, wird ein standardmäßiger, überwachter Ler-

ansatz namens Emotion Recognition Chain verwendet, der im folgendem Kapitel beschrieben wird.

3.6 Emotion Recognition Chain

Verantwortlich: Artur
- ERC Bild ändern (5 statt 4 Schritte)

Die Emotion Recognition Chain (ERC) besteht aus fünf Hauptschritten: Datenerfassung, Vorverarbeitung, Segmentierung, Merkmalsextraktion und Klassifizierung (vgl. Abb. 1). In den folgenden Unterkapiteln wird für jeden Schritt eine allgemeine Erklärung gegeben.

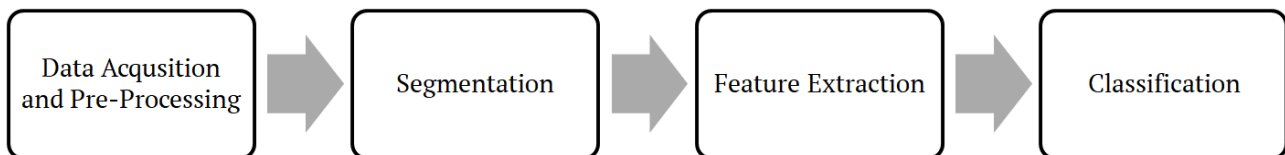


Abbildung 1: Emotion Recognition Chain: Zeitreihen-Datensätze werden von tragbaren Sensoren aufgenommen (Datenerfassung) und vorverarbeitet (Vorverarbeitung). Die Daten werden dann in Segmente unterteilt (Segmentierung), aus denen Merkmale extrahiert werden (Merkmalsextraktion). Mit den gewonnenen Merkmalen wird schließlich ein Klassifikator trainiert und anschließend dessen Ergebnisse bewertet (Klassifikation).

3.6.1 Datenerfassung

Verantwortlich: Artur
@Kevin, bitte Korrekturlesen und mit Deinem Teil abstimmen.

Dieser Schritt der ERC beinhaltet die Auswahl und den Aufbau der Sensoren, die Messreihendurchführung (um Daten zu erhalten) und Etikett-Beschriftungstechniken. Ziel ist es relevante und möglichst fehlerfreie Daten von Versuchspersonen für die verschiedenen emotionalen Zustände zu gewinnen. Der Datenerfassungsschritt ist besonders wichtig, da er der erste in der ERC ist und die Ergebnisse aller folgenden Schritte direkt von der Qualität des Datensatzes abhängen.

3.6.2 Vorverarbeitung

Verantwortlich: Artur
- Bereit zum Korrekturlesen.

Das Ziel der Vorverarbeitung ist die "Verbesserung" der Daten für die nachfolgenden Schritte der ERC. In der Regel ist es dadurch besser möglich Muster in Daten erkennen zu können. Vorverarbeitete Daten erreicht man durch Anwendung von z.B. Fil-

terung (Rauschunterdrückung), Normierung oder Reduzierung von unerwünschten oder unbedeutenden Datenteilen.

3.6.3 Segmentation

Verantwortlich: Artur
- Bereit zum Korrekturlesen.

Ziel dieses Schrittes ist es, Teile von Daten zu identifizieren, welche wichtige Informationen über die zu erkennenden Emotionen enthalten. Dies geschieht durch Filtern der Daten und Ausschließen von Segmenten, die für das Klassifizierungsproblem nicht relevant sind. Zusätzlich wird die zu verarbeitende Datenmenge reduziert, indem Segmente eines Zeitfensters fester Größe aus den Daten extrahiert werden. Diese Vorgehensweise ist heute in der Praxis besonders wichtig, da sonst hardwarebedingte Einschränkungen die zu verarbeitende Datenmenge begrenzen könnten.

3.6.4 Merkmalsextraktion

Verantwortlich: Artur
- Bereit zum Korrekturlesen.

Hier werden Charakteristiken und Merkmale in den Daten gesucht, die für das Klassifizierungsproblem von möglichst hoher Relevanz sind. Alle nach dem Segmentierungsschritt extrahierte Daten-Zeitfenster kann durch einen Merkmalsvektor (engl. "feature vector") dargestellt werden. Mit Hilfe von Merkmalsvektoren kann ein Klassifikator dann einfacher trainiert werden als nur mit den Rohdaten. Unser Fokus in der Mustererkennung lag vor allem auf der Merkmalsextraktion, da unsere Erfahrungen und frühere Forschungsarbeiten gezeigt haben, dass die Wahl der Merkmale sehr wichtig für die endgültigen Klassifizierungsergebnisse sind. Darüber hinaus wurden noch keine state-of-the-art Merkmale für die Emotionserkennung mit dieser spezifischen Assoziation von eindimensionalen physiologischen Signalen gefunden.

3.6.5 Klassifikation

Verantwortlich: Artur
- LOSOCV im späteren Kapitel erklären.
- Bereit zum Korrekturlesen.

Ziel des Klassifizierungsschrittes ist es, ein Klassifizierungsmodell zu trainieren, das in der Lage ist, Objekte in den Daten (dargestellt durch ihren Merkmalsvektor) in die entsprechende Klasse zuzuordnen.

Der Datensatz der Merkmalsvektoren, der im vorherigen Schritt des ERC erhalten wurde, wird in einen Trainingsset (engl. "training set") und einen Testset (engl. "testing set") unterteilt, so dass alle Klassen in beiden Sets vorhanden sind. Mit dem Trainingsdaten wird ein Klassifikator erstellt und trainiert. Der so erhaltene Klassifikator wird dann anhand der Daten des Testsets ausgewertet. Es ist wichtig, dass die Trainings- und Testsets unterschiedlich sind (d.h. nicht die gleichen im Daten Trainings- und Testset verwenden), da es sonst in einer Überanpassung (engl. "overfitting") des Klassifikator resultieren kann. Eine Überpassung tritt auf, wenn ein Klassifikator zufällige Schwankungen oder Rauschen in den Trainingsdaten "zu gut" lernt und dann bei neuen, unbekannten Daten deutlich schlechter abschneidet. Der Grund hierfür ist, dass diese gelernten Schwankungen oder Rauschen in den Trainingsdaten keinerlei Relevanz für das eigentliche Klassifizierungsproblem haben.

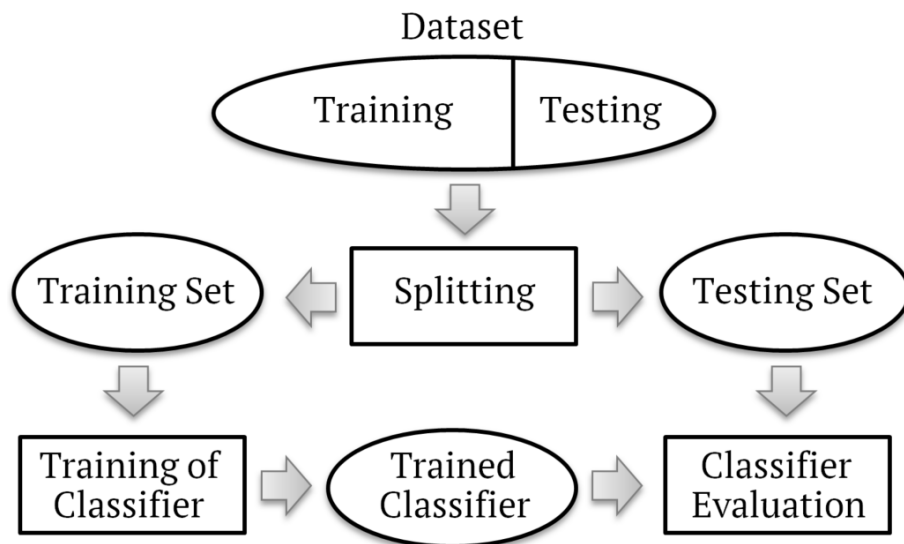


Abbildung 2: Aufteilen eines Datensets in ein Trainings- und ein Testset. Das Trainingsset dient zum Erstellen und Trainieren eines Klassifikators. Die Performance wird mithilfe des Testsets ermittelt und bewertet.

Für die Klassifizierung wurde der state-of-the-art Klassifikator Support-Vector-Machine (SVM) verwendet. SVM ist ein überwachtes Lernansatz, der für binäre Klassifikation oder Regressionszwecke genutzt werden kann. Das Ziel des SVM ist es eine Hyperebene (engl. "hyperplane") zu finden, die zwei Klassen im Raum der Merkmale trennt und gleichzeitig den Abstand (engl. "margin") zwischen der Hyperebene und den nächsten Datenpunkten jeder Klasse maximiert. Für jeden Datenpunkt (dargestellt durch den entsprechenden Vektor von Merkmalen) den wir klassifizieren möchten, wird ein Klassenetiket vergeben, je nachdem, zu welcher Seite der Hyperebene es gehört. Die Methode hat ihren Namen von den "Stützvektoren" (engl. "support vectors"), welche die nächstgelegenen Vektoren beider Klassen zur trennenden Hyperebene sind. Cortes und Vapnik zeigten in [2], dass die Gleichung der optimalen Hyperebene nur von diesen spezifischen Vektoren abhängig ist. Abbildung

3.6.5 zeigt eine optimale Hyperebene in 2D, die beide Klassen perfekt teilt. Datenpunkte werden durch nicht-ausgefüllte blaue Dreiecke bzw. rote Kreise für beide Klassen dargestellt, während Unterstützungsvektoren durch ausgefüllte Punkte und Kreise hervorgehoben werden.

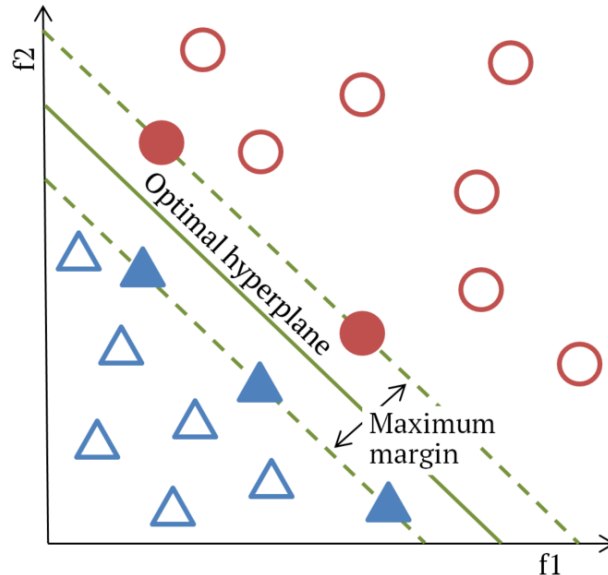


Abbildung 3: Beispiel eines SVM-Klassifikators im zweidimensionalen Merkmalsraum: Die Datenpunkte beider Klassen (dargestellt durch nicht-ausgefüllte blaue Dreiecke und rote Kreise) sind durch eine Hyper-ebene getrennt, welche die Margin maximiert. Stützvektoren werden als ausgefüllte Dreiecke und Kreise dargestellt.

Es ist anzumerken, dass SVM nur mit zwei Klassen funktioniert, aber sie können zu Z Klassen verallgemeinert werden. Es gibt zwei Ansätze zur Verallgemeinerung auf Z Klassen:

- 1 vs 1 besteht darin, Klassifikatoren für jedes Klassenpaar zu trainieren. Auf diese Weise werden $Z(Z - 1)/2$ Klassifikatoren trainiert, d.h. einen für jedes Klassenpaar.
- 1 vs all besteht darin, $Z - 1$ Klassen als eine Klasse zu betrachten und die letzte als zweite Klasse anzunehmen, mit der der Klassifikator trainiert wird. Dies wird für jede Klasse wiederholt, was zu Z Klassifikatoren führt, d.h. einen für jede Klasse.

In der Praxis sind die Daten durch normales SVM fast nie linear trennbar. Einer der Hauptgründe für die Beliebtheit von SVM ist jedoch die Möglichkeit es so genannten Kernel-Tricks. Es basiert auf der Annahme, dass nicht-linear trennbare Daten linear trennbar werden können, wenn sie in einen Raum höherer Dimension projiziert werden. In [2] zeigten Cortes und Vapnik, dass die SVM-Klassifikationsentscheidungsfunktion als gewichtete Summe von Skalarprodukten zwischen Stützvektoren und dem Vektor der zu klassifizierenden Merkmale ausgedrückt werden kann. Der Kernel-Trick

nutzt dies aus, indem er eine Kernelfunktion einführt, die ein skalares Produkt im hochdimensionalen Zielraum repräsentiert. Diese Kernelfunktion erübrigt die eigentliche Zuordnung zwischen dem ursprünglichen und dem hochdimensionalen Feature-Raum. Außerdem ist die Verwendung des Kernel-Tricks oft rechengünstiger als andere Alternativen. Die beiden beliebtesten Kernel sind der lineare und der Radial Basis Function (RBF) Kernel, definiert als:

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|) \quad (1)$$

wobei x und x' Vektoren des Merkmalsraums bezeichnen und γ der Parameter ist, der die "Ausbreitung" (engl. "Spread") des Kernels definiert.

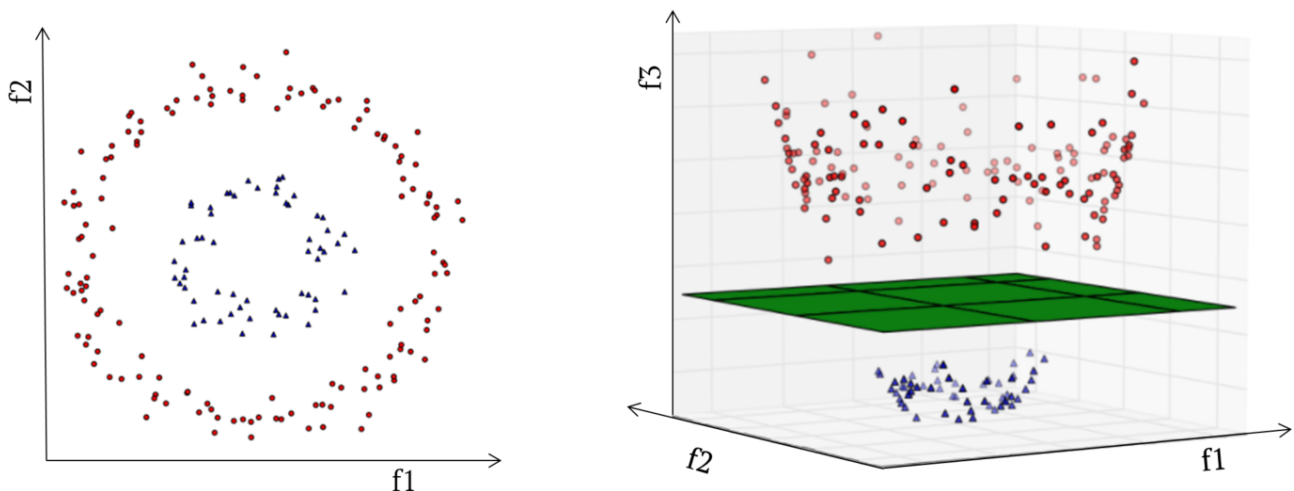


Abbildung 4: Beispiel für die Verwendung des Kernel-Tricks: Nicht-linear trennbare Daten in einem 2D-Merkmalraum (links) können linear trennbar gemacht werden, wenn sie auf 3D projiziert werden (rechts) [3].

Normales SVM verwendet sogenannte hard-margins, die versuchen, eine optimale Hyperebene zu schaffen, die keine Fehlklassifizierungen zulässt. Es ist jedoch oft besser, einige Klassifizierungsfehler zuzulassen, um eine Überanpassung zu verhindern und eine generalisierte Hyperebene zu erhalten. Diese allgemeinere Hyperebene liefert deutlich bessere und zuverlässigere Ergebnisse, wenn sie auf neue und unbekannte Datensätze angewendet wird. Hard-margin SVM kann zu einem Modell führen, das für die Trainingsdaten perfekt funktioniert, aber bei anderen Datensätzen sehr schlecht, weil es seine Trainingsdaten "zu gut" gelernt hat. Aus diesem Grund haben Cortes und Vapnik in [2] eine Variante des Standard-SVM-Klassifikators namens soft-margin SVM (oder auch C-SVM genannt) eingeführt, die eine Fehlklassifizierung von Beispielen beim Erstellen der trennenden Hyperebene toleriert. Der soft-margin Parameter C wird genutzt, um die Anzahl der Fehlklassifikationen festzulegen. Je größer der Wert von C , desto weniger Fehleinstufungen sind zulässig. Umgekehrt erlauben kleine Werte von C mehr Fehlklassifizierungen, um die Verallgemeinerungsfähigkeit des Klassifikators zu verbessern.

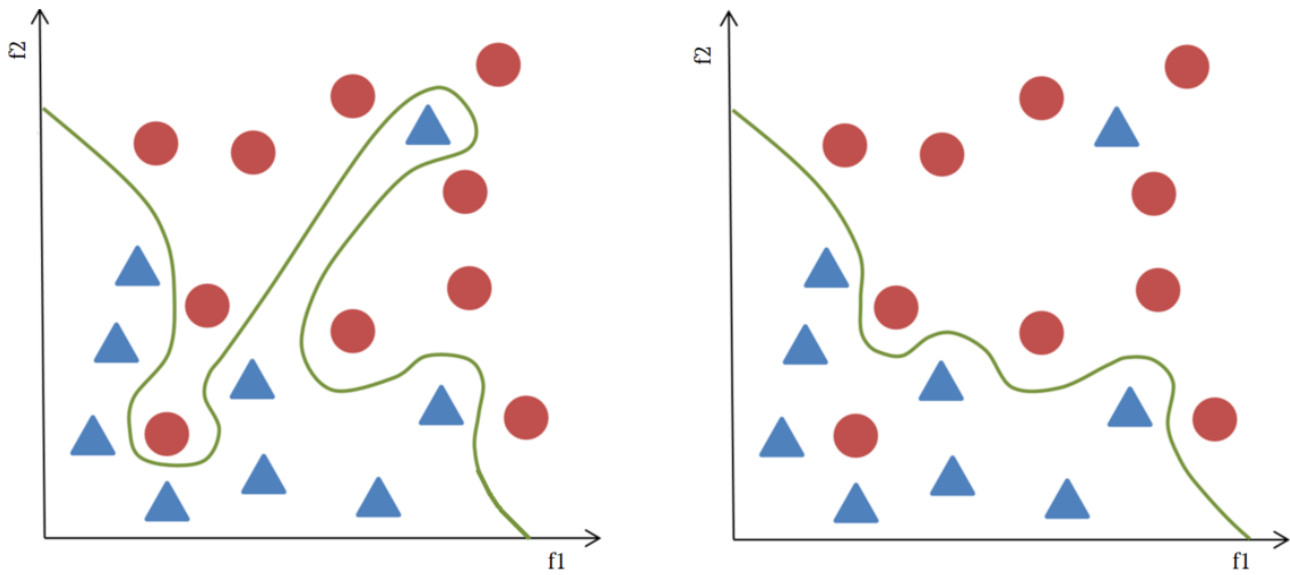


Abbildung 5: Beispiele für einen hard-margin SVM (links) und einen soft-margin SVM (rechts) in einem 2D-Merkmalraum: Der hard-margin SVM trennt die beiden Klassen perfekt im Gegensatz zum Soft-Margin SVM, der einige Fehlklassifikationen zulässt.

4 State-of-the-Art Analyse

Verantwortlich: Jonas

- Next Step: Gliederung erstellen

Part I

Erster Prototype

5 Systementwurf und Konzept

Verantwortlich: Kevin, Jonas

5.1 Anforderungen

5.2 Konzept

5.3 Hardwareauswahl

5.3.1 Auswahlkriterien

5.3.2 Festlegung der genutzten Hardware

5.4 Hardwarearchitektur

5.4.1 GSR-Sensor

5.4.2 Temperatur-Sensor

5.4.3 Pulsoximeter

5.4.4 EEG

5.4.5 EOG

5.4.6 Datenübertragung

5.5 Programmierung

5.6 Aufnahme der übertragenen Daten

6 Realisierung

Verantwortlich: Jonas, Kevin
- Next Step: Gliederung erstellen

7 Emotionsinduktion

Verantwortlich: Minas

7.1 Ablauf

Verantwortlich: Minas

7.2 Fragebogen

Verantwortlich: Boris

7.3 Szenarien

Verantwortlich: Meryem

7.3.1 Glück

Verantwortlich: Minas

7.3.2 Langeweile

Verantwortlich: Boris

7.3.3 Frustration

Verantwortlich: Meryem

8 Messreihe

Verantwortlich: Kevin, Artur

9 Mustererkennung

Verantwortlich: Artur

In diesem Kapitel werden alle Schritte entlang der ERC (vgl. Kapitel ??) konkretisiert und es wird genauer und detaillierter beschrieben wie vorgegangen wurde.

9.1 Datenerfassung

In Kapitel ?? wurde das verwendete Datenset bereits detailliert beschrieben, sodass hier darauf verzichtet wird.

9.2 Vorverarbeitung

Wie bereits in Kapitel ?? beschrieben, ist das Ziel der Vorverarbeitung die "Verbesserung" der Daten für die nachfolgenden Schritte der ERC. Im Rahmen des ELISE Projektes wurden Normalisierungstechniken auf dem gesamten Datensatz angewendet. Wir haben insbesondere die Standardnormalisierung verwendet, welche den Mittelwert der Daten auf Null setzt und die Einheitsvarianz ergibt [4]. Die Formel für die Standardnormierung lautet:

$$x' = \frac{x - \bar{x}}{\sigma} \quad (2)$$

wobei x ein Datenpunkt eines Sensorkanales, \bar{x} ist der Durchschnitt der Gesamtheit für diesen Sensorkanal und σ ist die entsprechende Standardabweichung.

9.3 Segmentation

Wie bereits in Kapitel ?? beschrieben, ist das Ziel der Segmentation Teile von Daten zu identifizieren, welche wichtige Informationen über die zu erkennenden Emotionen enthalten. In dieser Projektarbeit wurde ein Schiebefensteransatz (engl. "sliding window approach") verwendet. Ziel der Methode ist die Segmentierung der vorhandenen Daten in kleinere Einheiten, um die Merkmalsextraktion sowie die anschließende Klassifizierung zu vereinfachen oder gar erst zu ermöglichen. Die Länge des Zeitfensters (engl. "time window") und des Gleitschritts (engl. "sliding stride") sind zu bestimmende Parameter (und werden auch als "Hyperparameter" bezeichnet), wobei sich das Zeitfenster auf die feste Größe pro extrahiertem Segment und der Gleitschritt auf den Abstand zu dem Beginn des darauf folgenden Zeitfensters bezieht. Es ist zu beachten, dass sich aufeinanderfolgende Zeitfenster überlappen können, sobald der definierte Gleitschritt kleiner als das Zeitfenster ist.

Die Daten werden auf Zeitstempel-Ebene mit Etiketten beschriftet, basierend auf den von der jeweiligen Versuchsperson ausgefüllten Fragebögen. Jedem Zeitfenster wird ein Etikett zugeordnet, welches das dominante (d.h. am meisten vorhandene) Etikett der im entsprechenden Fenster enthaltenen Zeitstempel basiert. Es wird davon ausgegangen, dass jedes Zeitfenster nur von einer Emotion belegt ist.

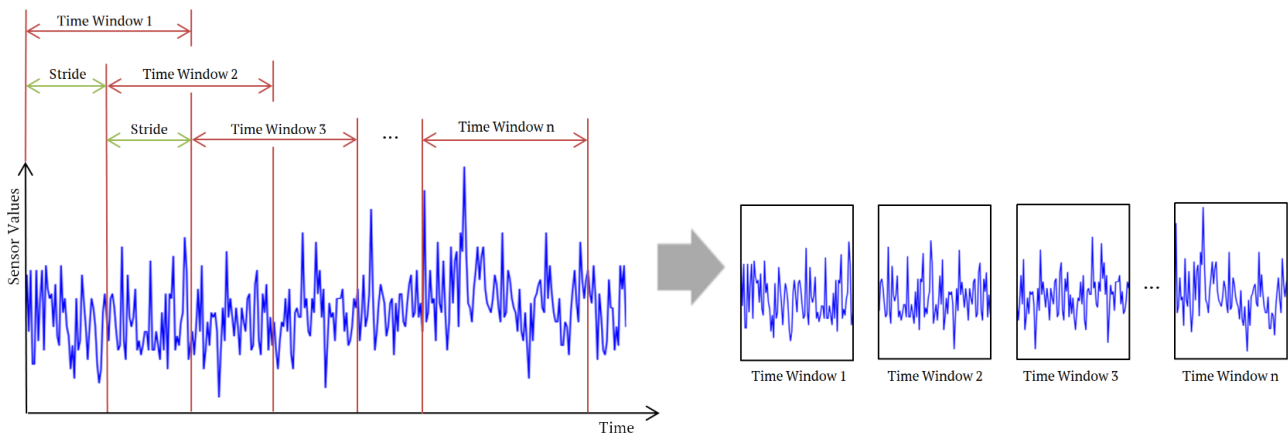


Abbildung 6: Schiebefenster-Segmentierung: Die Daten werden durch ein Zeitfenster fester Größe in kleinere Segmente aufgeteilt. Das Fenster wird mit einem festen Gleichschritt geschoben, um den aufeinanderfolgenden Daten-Zeitfenster zu erhalten.

9.4 Merkmalsextraktion

Wie bereits in Kapitel ?? beschrieben, ist das Ziel der Merkmalsextraktion Charakteristiken und Merkmale in den Daten zu finden, die für das Klassifizierungsproblem von möglichst hoher Relevanz sind. Im Rahmen des ELISE Projektes haben wir verschiedene Vorgehensweisen angewendet. Im den folgenden Unterkapiteln werden diese vorgestellt.

9.4.1 Handgefertigte Merkmale

Der handgefertigten Merkmal Ansatz (enlg. "hand-crafted features approach") besteht in der Berechnung relativ einfacher Merkmale von denen vermutet wird, dass sie für das Klassifizierungsproblem der Eingangssignale relevant sein können. Diese Vorgehensweise hat den Vorteil des einfachen Aufbaus als auch der relativ geringen benötigten Rechenleistung, wobei potentiell gute Klassifizierungsergebnisse erwarten werden.

Obwohl frühere Forschungsarbeiten schon handgefertigte Merkmale zur Emotionserkennung unter mithilfe physiologischer Signale getestet haben (vgl. [5]), wurde dieser Ansatz noch nie für die Erkennung dieser spezifischen Emotionen unter Verwendung dieser Kombination von Sensoren getestet. Zusätzlich haben wir zuerst

handgefertigte Merkmale getestet, um ein Basisergebnis zu liefern, mit der die Ergebnissen der anderen Ansätze verglichen werden können. Handgefertigte Merkmale sind in der Regel entweder einfache statistische Werte, Fourier-basierte oder selbstentwickelte Merkmale sein, die aufgrund von Vorkenntnissen der Daten verwendet werden. Diese Arbeit wurden statistische, Fourier-basierte und selbstentwickelte Merkmale getestet.

Statistische Merkmale

Die Tabelle 1 fasst die elf verschiedenen und in der Studie verwendeten statistischen Merkmale zusammen. Wir bezeichnen $\mathbf{x} = (x_1, x_2, \dots, x_T)$ als Vektor, der die in einem Datenzeitfenster der Länge T enthaltenen Sensorwerte für einen Sensorkanal darstellt.

Merkmalsname	Definition
Durchschnitt	$mean(\mathbf{x}) = \frac{1}{T} \sum_{k=1}^T (x_k)$
Standard-Abweichung	$\sigma(\mathbf{x}) = \sqrt{\frac{1}{T} \sum_{k=1}^T (x_k - \mu)^2}$
Maximum	$max(\mathbf{x}) = \max(x_1, x_2, \dots, x_T)$
Minimum	$min(\mathbf{x}) = \min(x_1, x_2, \dots, x_T)$
Amplitude	$A(\mathbf{x}) = max(\mathbf{x}) - min(\mathbf{x})$
25/50/75% Perzentil	Wert einer Menge, unter dem 25/50/75% der Werte aus der Menge fallen.
Interquartiler Bereich	Differenz zwischen dem 75. und 25. Perzentil.
Schräge	$\gamma_1(\mathbf{x}) = E \left[\left(\frac{X-\mu}{\sigma} \right)^3 \right] = \frac{\mu_3}{\sigma^3} = \frac{E[(X-\mu)^3]}{(E[(X-\mu)^2])^{3/2}} = \frac{\kappa_3}{\kappa_2^{3/2}}$
Kurtosis	$Kurt[\mathbf{x}] = E \left[\left(\frac{X-\mu}{\sigma} \right)^4 \right] = \frac{\mu_4}{\sigma^4} = \frac{E[(X-\mu)^4]}{(E[(X-\mu)^2])^2}$

Tabelle 1: Statistische Merkmale, die im Rahmen des ELISE-Projektes verwendet wurden.

Fourier-basierte Merkmale

Artur:

- Muss ich noch schreiben → Warten auf Julian's Antwort.

Selbstentwickelte Merkmale

Es wurden zwei eigene Merkmale definiert: Nulldurchgang (engl. "zero crossing") und Anzahl der Spitzen (engl. "number of peaks"). Im Folgendem werden diese beiden Merkmale detailliert beschrieben.

Das Nulldurchgang-Merkmal zählt die Häufigkeit, mit der das Signal eines Sensorkanals in einem Zeitfenster die Nulllinie überschreitet. Alle Sensorsignale wurden durch Normierung verarbeitet (vgl. Kapitel xxx) und damit wurden alle Mittelwerte auf Null zentriert. Um zu vermeiden, dass Rauschen entlang der Nulllinie in dem Merkmal gezählt wird, wird nur ein Nulldurchgang in einer bestimmten Zeitspanne gezählt.

Das Spitzenzähler-Merkmal bestimmt die Anzahl von lokalen Hochpunkten im Zeitsignal. Alle lokalen Maximen sind durch einen Onset (Startpunkt), eine Spitze und einen Offset (Endpunkt) gekennzeichnet (vgl. Abbildung 7). Jedes Vorkommen einer Onset/Offset-Paarung wird hierbei als Spitze gezählt. Onsets, Spitzen und Offsets werden durch die folgenden Operationen identifiziert [?]:

- Ein Onset wird bestimmt, wenn der Wert des Signals an diesem Punkt nicht negativ ist und die Differenz zwischen ihm und dem nächsten größer als ein vordefinierter Schwellenwert (engl. "threshold") ist.
- Ein Offset wird bestimmt, wenn der Wert des Signals kleiner als der Wert des zuletzt gesetzten Onsets ist.
- Das lokale Maximum zwischen einem Onset und Offset wird als Spitze bezeichnet.

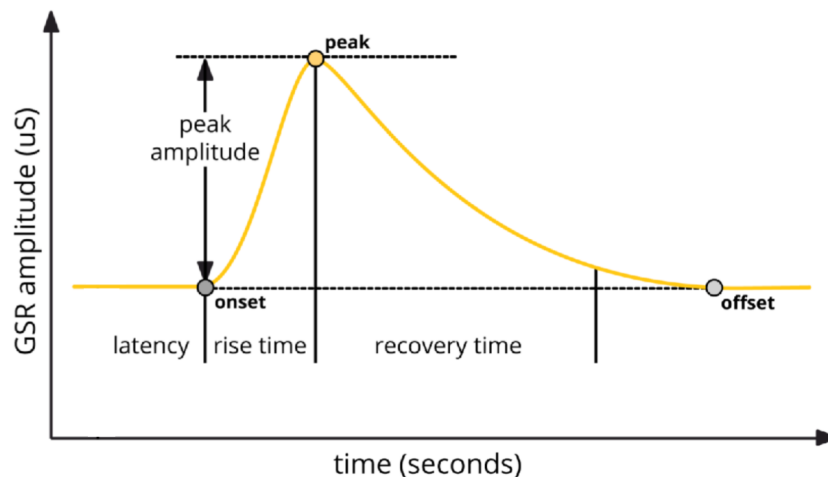


Abbildung 7: Spitzenzähler: Onset (Startpunkt), Spitze und Offset (Endpunkt). Jedes Paar von Onset/Offset erhöht die Anzahl der Spitzen um eins.

Jedes handgefertigte Merkmal wird auf einem Zeitfenster von Daten für jeden Sensorkanal unabhängig voneinander angewendet. Jedes Zeitfenster ist daher 117 Merkmalen zugeordnet (9 Sensorkanäle multipliziert mit 13 Merkmalen).

9.4.2 Codebook Approach

Die bisherige Methodik mit handgefertigten Merkmalen ist klassisch für den überwachten Lernansatz. Es existieren aber auch einige Nachteile. Das Hauptproblem besteht darin, dass nicht sichergestellt werden kann, dass die gewählten Merkmale die besten Klassifizierungsergebnisse erzielen. Damit besteht immer die Gefahr, dass möglicherweise andere Merkmale bessere Ergebnisse liefern würden, diese handgefertigten Merkmale aber nicht gefunden wurden. Dieses Risiko besteht insbesondere bei der physiologischen Signalverarbeitung zur Emotionserkennung, wo die Struktur der Daten noch recht unbekannt und allgemein komplex ist. Eine weitere Schwierigkeit besteht darin, relevante selbstentwickelte Features ohne Expertenwissen über die Daten zu finden. Darüber hinaus wurden noch keine gut funktionierenden State-of-the-Art handgefertigten Merkmale identifiziert. Aus diesen Gründen ist es interessant halbautomatische und unüberwachter Ansätze der Merkmalsextraktion zu verwenden und zu testen.

K. Shirahama et al. [6] schlugen eine unüberwachte Merkmalsextraktionsmethode namens Codebook Approach (CA) vor, um Merkmale aus 1D-Zeitreihensignalen zu erzeugen. Der CA hat den Vorteil, dass formbasierte Merkmale gefunden werden können, die für das Problem der Emotionserkennung relevant sind, aber weder offensichtlich noch leicht als Mensch zu interpretieren sind. Der CA besteht aus drei Schritten, die in den folgenden Abschnitten erläutert werden: Codebuchkonstruktion (engl. "codebook construction"), Codewortzuordnung (engl. "codeword assignment") und der anschließenden Klassifizierung.

Codebuchkonstruktion

Ziel dieses Schrittes ist es, Teilsequenzen (sogenannte "Codewörter") zu bestimmen, die für die 1D-Eingangssensorik charakteristisch sind. Dies wird erreicht, indem Zeitfenster aus dem ursprünglichen Datensatz für jeden Sensorkanal unabhängig voneinander nach dem im Kapitel 9.3 definierten Segmentierungsansatz extrahiert werden. Aus jedem so erhaltenen Zeitfenster der Größe T werden kleinere Segmente der Größe α unterteilt. Ein Clustering-Algorithmus wird dann auf die Menge der Segmente α angewendet, um Clusterzentren zu finden. Nach der Konvergenz werden die Clusterzentren als Codewörter betrachtet und zum Aufbau einer Sammlung von Codewörtern mit dem Namen "Codebuch" verwendet, wie in Abbildung 8 aus [6] dargestellt. Die Anzahl der Codewörter (d.h. die Größe des Codebuchs oder die Anzahl der Cluster) ist ein Hyperparameter des Verfahrens. Im Rahmen dieser Arbeit wurde ein k-means Clustering-Algorithmus verwendet, um die Codewörter auf den ELISE-Daten zu erhalten.

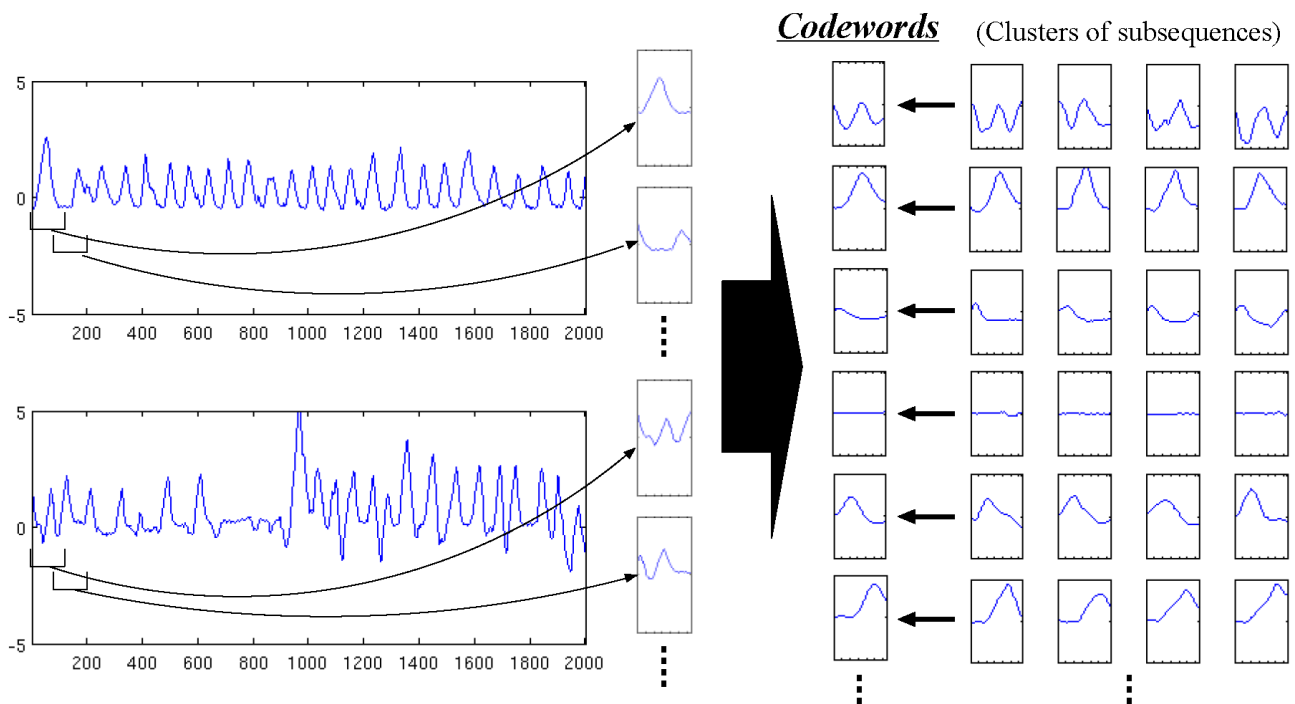


Abbildung 8: Codebuchkonstruktion: Zeitfenster von Daten werden zunächst extrahiert. Ein Clustering-Algorithmus wird dann auf das alle Zeitfenster angewendet, um Clusterzentren (und damit Codewörter) zu finden, die zum Aufbau des Codebuchs verwendet werden.

Codewortzuordnung

Nach der Konstruktion der Codewörter wird für jedes Zeitfenster T ein histogrammbasierter Merkmalsvektor erstellt (vgl. Abbildung 9, entnommen aus [6]). Der zu klassifizierende Datensatz wird zunächst in Zeitfenster der Größe T segmentiert, aus denen Segmente der Größe α nach dem gleichen Verfahren wie beim Aufbau des Codebuchs extrahiert werden. Jedes Segment α wird dann mit den Codewörtern verglichen, so dass das "ähnlichste" Codewort gefunden werden kann. Ein K-Bin-Histogramm (mit K Anzahl der Codewörter) mit Informationen über die Anzahl der Male enthält, die jedes Codewort als am ähnlichsten zu den Segmenten α im Zeitfenster T betrachtet wurde, wird dann erstellt und als Merkmalsvektor verwendet, um das zu klassifizierende Zeitfenster T darzustellen. Das Maß für die Ähnlichkeit von Codewörtern und Datensegmenten α basiert auf der euklidischen Entfernung.

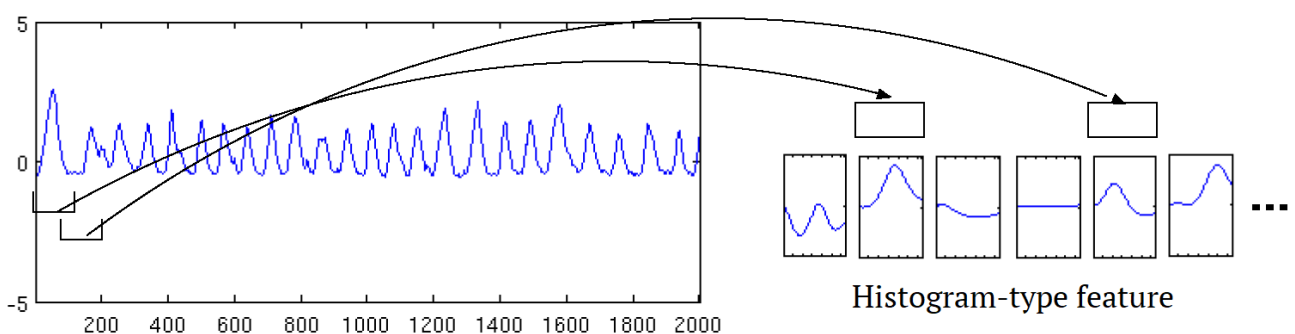


Abbildung 9: Codewortzuweisung: Jedes Datensegment wird mit den Codewörtern verglichen und ein Histogramm mit Informationen darüber, wie oft jedes Codewort als "am ähnlichsten" betrachtet wurde, wird erstellt.

Der zuvor beschriebene Ansatz wird als "Hard-Zuordnung" (engl. "hard assignment") bezeichnet, da Datensegmente einem einzigen Codewort zuzuordnen werden. Ein Nachteil dieser Vorgehensweise ist die mangelnde Flexibilität im Umgang mit potenzieller Unsicherheit bei der Codewortzuweisung, die z.B. auftreten kann, wenn ein Zeitfenster von Daten zwei oder mehr Codewörtern sehr ähnlich ist, da es nur einem zugeordnet werden kann. Ein alternativer Ansatz dieses Problem zu umgehen ist die so genannte "Soft-Zuordnung" (engl. "soft assignment"). Hierbei werden auch alle Codewörter des Codebuchs Datensegmenten zuzuordnen, wobei die Ähnlichkeit jeweils als ein Bin im Histogramms dargestellt wird. Sehr ähnlich entspricht hierbei einem hohen Wert und nicht ähnlich einem kleinen Wert (anstatt 0 oder 1 wie in der Variante der Hard-Zuordnung). Eine Kerneldichtefunktion (vgl. [7]) wird verwendet, um die Histogramm-Bins zu berechnen. Es wurde die folgende Funktion verwendet:

$$f(\alpha, c_k, p) = \frac{1}{\gamma} \frac{g(\alpha, c_k, p)}{\sum_{i=1}^K g(\alpha, c_i, p)} \quad (3)$$

wobei $f(\alpha, c_k, p)$ die Ähnlichkeit des Segments α bezeichnet, das sich auf das Codewort c_k bezieht. p ist ein Glättungsparameter (ein großes p bewirkt eine starke Glättung), γ ist die Anzahl der Segmente im Zeitfenster T und:

$$g(\alpha, c_k, p) = \frac{1}{\sqrt{2\pi p^2}} \exp\left(-\frac{d(\alpha, c_k)^2}{2p^2}\right). \quad (4)$$

Um numerischen Unterlauf zu vermeiden, wird die Gleichung (4) zunächst mit dem Log-Sum-Exp-Trick (vgl. [8]) berechnet.

Fusion mehrerer Sensoren

Oftmals werden mehrere Sensoren verwendet, die gleichzeitig mehrere verschiedene Signale derselben Emotion erzeugen. Die Fusion dieser Signale ist wichtig, da sie die Genauigkeit der Emotionserkennung verbessern kann. Es gibt zwei verschiedene Ansätze (vgl. [9]): die frühe Fusion (engl. "early fusion") und die späte Fusion (engl. "late fusion"):

- Frühe Fusion: In der Dimension $K \times S$ wird nur ein Klassifizierer benötigt, wobei K die Anzahl der Codewörter und S die Anzahl der Sensorkanäle ist. Der Klassifikator wird anhand der Verkettung von Codebuchmerkmalen trainiert und ausgewertet, die auf jedem Sensorkanal unabhängig voneinander berechnet wurden.
- Späte Fusion: Erfordert mindestens S -Klassifikatoren (einen für jeden Sensorkanal). Ein Klassifizierer wird unabhängig für jeden Sensorkanal unter Verwendung der für den betrachteten Sensor erhaltenen Codebuchmerkmale trainiert.

Die Vorhersagen der \mathcal{S} -Klassifikatoren werden dann fusioniert, um die Klassenbezeichnung des zu klassifizierenden Zeitfensters zu schätzen (z.B. mit einem zusätzlichen Klassifizierer).

In dem ELISE Projekt verwenden wir den späten Fusionsansatz, weil er rechnerisch günstiger ist und von K. Shirahama (vgl. [6]) empfohlen wird.

9.5 Klassifikation

10 Ergebnisse

Verantwortlich: Artur

10.1 Ergebnisse der hand-gefertigten Merkmale

10.2 Ergebnisse des Codebook Approach

10.3 Analyse der Ergebnisse

Part II**Zweiter Prototype****11 Systementwurf und Konzept**

Verantwortlich: Kevin, Jonas

11.1 Anforderungen**11.2 Konzept****11.3 Hardwareauswahl****11.3.1 Auswahlkriterien****11.3.2 Festlegung der genutzten Hardware****11.4 Hardwarearchitektur****11.4.1 GSR-Sensor****11.4.2 Temperatur-Senosr****11.4.3 Pulsoximeter****11.4.4 EEG****11.4.5 EOG****11.4.6 Datenübertragung****11.5 Programmierung****11.6 Aufnahme der übertragenen Daten**

12 Realisierung

Verantwortlich: Jonas, Kevin
- Next Step: Gliederung erstellen

13 Emotionsinduktion

Verantwortlich: Minas

13.1 Ablauf

Verantwortlich: Minas

13.2 Fragebogen

Verantwortlich: Boris

13.3 Szenarien

Verantwortlich: Meryem

13.3.1 Glück

Verantwortlich: Minas

13.3.2 Langeweile

Verantwortlich: Boris

13.3.3 Frustration

Verantwortlich: Meryem

14 Messreihe

Verantwortlich: Kevin, Artur

15 Mustererkennung

Verantwortlich: Artur

15.1 Merkmalsextraktion für Emotionserkennung

15.1.1 Hand-gefertigte Merkmale

15.1.2 Codebook Approach

16 Ergebnisse

Verantwortlich: Artur

16.1 Ergebnisse der hand-gefertigten Merkmale

16.2 Ergebnisse des Codebook Approach

16.3 Analyse der Ergebnisse

Part III**Dritter Prototype****17 Systementwurf und Konzept**

Verantwortlich: Kevin, Jonas

17.1 Anforderungen**17.2 Konzept****17.3 Hardwareauswahl****17.3.1 Auswahlkriterien****17.3.2 Festlegung der genutzten Hardware****17.4 Hardwarearchitektur****17.4.1 GSR-Sensor****17.4.2 Temperatur-Sensor****17.4.3 Pulsoximeter****17.4.4 EEG****17.4.5 EOG****17.4.6 Datenübertragung****17.5 Programmierung****17.6 Aufnahme der übertragenen Daten**

18 Realisierung

Verantwortlich: Jonas, Kevin
- Next Step: Gliederung erstellen

19 Emotionsinduktion

Verantwortlich: Minas

19.1 Ablauf

Verantwortlich: Minas

19.2 Fragebogen

Verantwortlich: Boris

19.3 Szenarien

Verantwortlich: Meryem

19.3.1 Glück

Verantwortlich: Minas

19.3.2 Langeweile

Verantwortlich: Boris

19.3.3 Frustration

Verantwortlich: Meryem

20 Messreihe

Verantwortlich: Kevin, Artur

21 Mustererkennung

Verantwortlich: Artur

21.1 Merkmalsextraktion für Emotionserkennung

22 Ergebnisse

Verantwortlich: Artur

22.1 Ergebnisse der hand-gefertigten Merkmale

22.2 Ergebnisse des Codebook Approach

22.3 Analyse der Ergebnisse

Part IV**Vierter Prototype****23 Systementwurf und Konzept**

Verantwortlich: Kevin, Jonas

23.1 Anforderungen**23.2 Konzept****23.3 Hardwareauswahl****23.3.1 Auswahlkriterien****23.3.2 Festlegung der genutzten Hardware****23.4 Hardwarearchitektur****23.4.1 GSR-Sensor****23.4.2 Temperatur-Sensor****23.4.3 Pulsoximeter****23.4.4 EEG****23.4.5 EOG****23.4.6 Datenübertragung****23.5 Programmierung****23.6 Aufnahme der übertragenen Daten**

24 Realisierung

Verantwortlich: Jonas, Kevin
- Next Step: Gliederung erstellen

25 Emotionsinduktion

Verantwortlich: Minas

25.1 Ablauf

Verantwortlich: Minas

25.2 Fragebogen

Verantwortlich: Boris

25.3 Szenarien

Verantwortlich: Meryem

25.3.1 Glück

Verantwortlich: Minas

25.3.2 Langeweile

Verantwortlich: Boris

25.3.3 Frustration

Verantwortlich: Meryem

26 Messreihe

Verantwortlich: Kevin, Artur

27 Mustererkennung

Verantwortlich: Artur

27.1 Merkmalsextraktion für Emotionserkennung

28 Ergebnisse

Verantwortlich: Artur

28.1 Ergebnisse der hand-gefertigten Merkmale

28.2 Ergebnisse des Codebook Approach

28.3 Analyse der Ergebnisse

29 Zusammenfassung und Ausblick

Machen wir später.

29.1 Zusammenfassung

29.2 Fazit

29.3 Ausblick

Abbildungsverzeichnis

1	Emotion Recognition Chain	13
2	Aufteilen eines Datensets in ein Trainings- und Testset	15
3	Beispiel eines SVM-Klassifikators im zweidimensionalen Merkmal- raum	16
4	Beispiel für die Verwendung des Kernel-Tricks	17
5	Beispiele für einen hard-margin SVM (links) und einen soft-margin SVM (rechts) in einem 2D-Merkmalsraum	18
6	Schiebefenster-Segmentierung	25
7	Spitzenzähler: Onset (Startpunkt), Spitze und Offset (Endpunkt). Jedes Paar von Onset/Offset erhöht die Anzahl der Spitzen um eins.	27
8	Codebuchkonstruktion: Zeitfenster von Daten werden zunächst ex- trahiert. Ein Clustering-Algorithmus wird dann auf das alle Zeit- fenster angewendet, um Clusterzentren (und damit Codewörter) zu finden, die zum Aufbau des Codebuchs verwendet werden.	29
9	Codewortzuweisung: Jedes Datensegment wird mit den Codewörtern verglichen und ein Histogramm mit Informationen darüber, wie oft jedes Codewort als "am ähnlichsten" betrachtet wurde, wird erstellt.	29

Tabellenverzeichnis

1	Statistische Merkmale, die im Rahmen des ELISE-Projektes verwendet wurden.	26
---	------------------------------------------------------------------------------------	----

Abkürzungen

Bitte alle verwendete Abkürzungen nochmals hier aufführen.

ANN	Artificial Neural Networks
BMBF	Bundesministerium für Bildung und Forschung
BVP	Blood Volume Pulse
CA	Codebook Approach
CRID	Center for Responsible Innovation & Design
C-SVM	Soft-margin Support-Vector-Machine
CSV	Comma-Separated-Values
EEG	Electroencephalography
EOG	Electrooculography
ERC	Emotion Recognition Chain
GSR	Galvanic Skin Response
HR	Heart Rate
LOSOVCV	Leave-One-Subject-Out-Cross-Validation
PG	Projektgruppe
PPG-ir	Photoplethysmography Infrared
PPG-red	Photoplethysmography Red
RBF	Radial Basis Function
SpO2	Pulse Oximetry
SVM	Support-Vector-Machine
VR	Virtual Reality

Anhang

Referenzen

- [1] X. Zhu. Semi-supervised learning literature survey, July 2008.
- [2] C. Cortes and V. Vapnik. *Support-Vector Networks*, volume volume 20. Kluwer Academic Publishers, Boston, September 1995.
- [3] E. Kim. Everything you wanted to know about the kernel trick. September 2013.
- [4] J. Grus. *Data Science from Scratch*. O'Reilly Media, April 2015.
- [5] H.P. Martinez, Y. Bengio, and G.N. Yannakakis. Learning deep physiological models of affect. *IEEE Computational Intelligence Magazine*, volume 8(issue 2):pages 20–33, April 2013.
- [6] K. Shirahama, L. Koeping, and M. Grzegorzec. Codebook approach for sensor-based human activity recognition. *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 197–200, September 2016.
- [7] J.C. van Gemert, C.J. Veenman, and A.W.M. Smeulders. Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 32(issue 7):pages 1271–1283, June 2009.
- [8] K.P. Murphy. *Machine Learning and A Probabilistic Perspective*. The MIT Press, London, September 2012.
- [9] C.G.M. Snoek, M. Worring, and A.W.M. Smeulders. Early versus late fusion in semantic video analysis. *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 399–402, November 2005.