Institut für Telematik
Technische Universität Hamburg

Forschungsprojekt und Seminar

Computer Science

Handgestenerkennung mit Entscheidungsbäumen

von

Tom Dymel

Februar 2021

Betreut von

Dr. Venzke
Institute of Telematics, Hamburg University of Technology

Erstprüfer	Prof. Dr. Volker Turau Institute of Telematics Hamburg University of Technology
Zweitprüfer	Dr. Venzke Institute of Telematics Hamburg University of Technology

Inhaltsverzeichnis

Kapitel 1

Einleitung

Machinelles Lernen (ML) gewann in den vergangenen Jahren an Popularität, u.a. durch die Fortschritte in parallelen Rechnen, sinkende Speicherpreise, schnelleren Speicher und den Zugang zu Bibliotheken, wie zum Beispiel Scikit-Learn, Keras und PyTorch, welche den Einstieg in maschinellen Lernen erleichtern (TODO: Quelle?). Ein namenhaftes Beispiel für das Potential von maschinellen Lernen ist die AlphaGo-KI, die einen Sieg gegen den besten menschlichen Spieler erringen konnte in dem Spiel Go, welches als besonders schwierig für Computer zu meistern galt durch den enormen Suchraum von möglichen Aktionen (TODO: Quelle).

Ein häufiges Anwendungsgebiet in eingebetteten Systemen ist die optische Gestenerkennung, die zur kontaktlosen Interaktion mit technischen Geräten u. a. genutzt wird (Todo: Quelle). Die eingesetzten Micro-Controller sind jedoch häufig nicht ausreichend leistungsstark, um ein trainiertes Model in passabler Zeit auszuführen (TODO: Quelle). Gründe dafür sind Kosten oder Anforderungen an die Batterielanglebigkeit (TODO: Quelle). Eine Lösung ist das Auslagern der Modelle in einen leistungsstarken Rechenkluster, indem die nötigen Eingabedaten auf dem Mikro-Controller gesammelt werden und anschließend an den Rechenkluster gesendet werden (TODO: Quelle). Dies erzeugt allerdings Latenz und eine Abhängigkeit zu einer solchen Infrastruktur. Ein alternativer Ansatz ist das lokale Ausführen der Modelle, was allerdings die Komplexität des Modells limitiert um eine passable Ausführungszeit zu gewährleisten.

In dieser Arbeit wird die Handgestenerkennung mit Entscheidungsbäumen auf dem Arduino-Board ATmega328P untersucht, das mit 9 Fotowiderständen ausgestattet ist, die in einer 3x3 Matrix angeordnet sind. Der ATmega328P verfügt über eine 8-Bit APU, 2 kB RAM, 32 kB Flash-Speicher und operiert unter 16 MHz. Im Vergleich zu vorherigen Arbeiten, die sich mit Künstlichen Neuronalen Netzen (KNN) auseinander gesetzt haben, versprechen Entschei-

dungsbäume einen geringeren Rechenaufwand.

Um dieses Ziel zu erreichen, werden Entscheidungsbaum basierte Klassifizierer auf einem leistungsstarken Computer trainiert und anschließend in eine bestehende Infrastruktur eingebettet, die Gestenkandidaten als Folge von Bildern identifiziert. Es werden insgesamt zwei Komponenten hinzugefügt. Die erste Komponente extrahiert Features, welche an die zweite Komponente, der Klassifizierer, weitergegeben werden um die Handgesten zu erkennen. Der Klassifizierer wird mit den vorherigen Arbeiten verglichen im Hinblick auf Ausführungszeit, Resourcenverbrauch und Erkennungsgenauigkeit unter verschiedenen Verhältnissen, wie Geschwindigkeit, Licht und Entfernung. Zusätzlich wird untersucht inwiefern `Nullgesten`, i. e. invalide Gesten, erkannt werden können und welche Konsequenzen es auf die Erkennungsgenauigkeit hat.

Zunächst werden in Kapitel 2 Entscheidungsbäume eingeführt, verschiedene Ensemble-Methoden erläutert und auf die Generierung von den Modellen eingegangen. In Kapitel 3 wird erläutert, wie Gestenkandidaten extrahiert werden und welche Ansätze vorherige Arbeiten bereits verfolgt hatten. Anschließend präsentiert Kapitel 4 den Kern der Arbeit. Zuallererst wird die Infrastruktur vorgestellt, welche im Rahmen dieser Arbeit angefertigt wurde, um Klassifizierer zu trainieren und zu evaluieren. Anschließend wird das Datenset `Dymel` vorgestellt, was zusätzlich zu dem bestehenden Datenset von `Klisch` erstellt wurde, um Nullgesten zu und verschiedene Helligkeitsstufen zu untersuchen. Danach werden die Features vorgestellt, die im Laufe dieser Arbeit untersucht wurden. Zuletzt wird die Erkennungsgenauigkeit, die Ausführungsgeschwindigkeit und der Resourcenverbrauch evaluiert. In Kapitel 5 werden die Schlussfolgerungen dargestellt.

Kapitel 2

Methoden

2.1 Entscheidungsbäume

Der Entscheidungsbaum ist eine rekursive Datenstruktur um Entscheidungsregeln darzustellen. Jedem Blatt ist eine Klasse zugeordnet und allen anderen Knoten ist ein *Test* zugeordnet. Der Test hat eine Reihe von sich gegenseitig ausschließenden Ergebnissen. Die Zuordnung einer Klasse zu einem Objekt wird durch das Traversieren dieses Baumes bestimmt bis ein Blatt erreicht wird [Qui90].

2.1.1 Konstruktion

Es gibt viele verschiedene Algorithmen um Entscheidungsbäume zu erzeugen: `ID3`, `C4.5`, `C5`, `CART`, `CHAID`, `QUEST`, `GUIDE`, `CRUISE` and `CTREE`. Am häufigsten wird ID3 (Iterative Dichotomizer 3), bzw. C4.5, welches eine Weiterentwicklung von ID3 ist, und CART (Classification and Regression Trees) verwendet [PVG+11, SG14]. Diese Arbeit verwendet die Implementierung für Entscheidungsbaumklassifizierer von *Scikit-Learn*, die eine optimierte Version von CART nutzt [Ent20]. Scikit-Learn ist ein Python-Modul, das eine große Auswahl von Algorithmen zum maschinellen Lernen implementiert [PVG+11].

CART: Classification and Regression Trees

Die Konstruktion eines optimalen binären Entscheidungsbaum ist NP-Komplett [LR76]. CART ist ein Greedy-Algorithmus, der lokal immer die beste Teilung wählt.

```
    Weise dem Wurzelknoten alle Trainingsdaten zu.
    Definiere den Wurzelknoten als Blatt.
    WHILE True:
        Neue_Teilungen = 0
        FOR jedes Blatt:
            IF die Größe der zugewiesenen Trainingsdaten zu klein ist oder alle ↙
                Einträge der Trainingsdaten zur gleichen Klasse gehören:
```

```
            CONTINUE
        Finde das Attribut, das am besten den Knoten in zwei Kindesknoten ↙
            unterteilt mit einer erlaubten Teilungsregel.
        Neue_Teilungen += 1
    IF Neue_Teilungen == 0:
        break
```

Listing 2.1: Skizze von vereinfachten Baumwachstumsalgorithmus [Ste09].

Listing 2.1 skizziert wie CART initial einen maximal großen Baum generiert indem die Trainingsdaten solange geteilt werden bis keine weitere Teilung mehr möglich ist oder alle Einträge der gleichen Klasse zugeordnet sind. Zuletzt beginnt der Reduzierungsprozess indem Teilbäume gelöscht werden, die die Klassifizierungsgenauigkeit nicht erhöhen oder der Zuwachs unter einem Nutzer definierten Schwellenwert liegt [Ste09].

2.2 Ensemble

2.2.1 Voting

2.2.2 Bagging

2.2.3 Random Forest

2.2.4 Boosting?

2.3 Cherry-Picking

Kapitel 3

State of the Art

3.1 Gesture candidate

3.2 Feed Forward Neural Network

3.2.1 Performance

3.2.2 Resource usage

3.2.3 Optimization

Compression

Pruning

3.3 Recurrent Neural Network

3.4 Related Work

TODO!

Kapitel 4

Achievements

4.1 Infrastructure

4.1.1 Parsing gestures

4.1.2 Feature extraction

4.1.3 Training

Scikit-Learn

Ensemble

4.1.4 Arduino

Export Tree as C code

Feature extraction

4.1.5 Utility Tools

Gesture Recorder

Playground

Gesture Simulation

4.1.6 Workflow?

4.2 DymelData

TODO: Motivation @ consistent real world evaluation

4.2.1 Configurations

4.2.2 NullGestures

Explain what it is in general

Corner

Same side in and out

Rotations

4.2.3 Synthetic Brightness Dataset

4.3 Features

4.3.1 Requirements

Invariance to velocity

Development over time

Sense of direction

Sense of position

4.3.2 Curse of dimensionality

4.3.3 Unpromising features

Explain that they had been tried out and did work out. Explain why

4.3.4 Brightness distribution

Explain whaat it is. Note this as "discretisized"position. Explain what requirements it fullfills. Explain different variants.

4.3.5 Motion History

Explain origin and how it works. Explain what requirements it fullfills and why.

Implementation

4.3.6 Center of Gravity Distribution

Motivate.?

Center of Gravity

Integer variant

Adding development over time

Window size

Also talk about the squishing?

4.4 Performance Evaluation

Talk about what was searched for, i.e. not all sizes had been tried out. Talk about different testset sizes? Talk about the greedy nature of the training algorithm? (Was that done in Cherry Picking?)

4.4.1 Testsets

Motivate. Explain what they are used for. Explain whats in there?

Metrics

Talk about used metrics

Klisch

DymelNull

DymelGesture

4.4.2 Feasible solution

4.4.3 Considering NullGestures

4.4.4 Brightness Distribution

Show graphs about: Best solution, Best feasible solution, With and WithOUT considering null gestures. Talk when it starts to generalize more poorly(?)

4.4.5 Motion History

Show graphs about: Best solution, Best feasible solution, With and WithOUT considering null gestures. Talk when it starts to generalize more poorly(?)

4.4.6 Brightness Distribution and Motion History

Show graphs about: Best solution, Best feasible solution, With and WithOUT considering null gestures. Talk when it starts to generalize more poorly(?)

4.4.7 Center of Gravity Distribution Float Ansatz

Show graphs about: Best solution, Best feasible solution, With and WithOUT considering null gestures. Talk when it starts to generalize more poorly(?)

Talk about brightness distribution!!

4.4.8 Center of Gravity Distribution Integer Ansatz

Show graphs about: Best solution, Best feasible solution, With and WithOUT considering null gestures. Talk when it starts to generalize more poorly(?)

Talk about brightness distribution!!

4.4.9 Center of Gravity Distribution Float and Integer Ansatz

Show graphs about: Best solution, Best feasible solution, With and WithOUT considering null gestures. Talk when it starts to generalize more poorly(?)

Here state what was expected and state how it worked out! Talk about brightness distribution!! ==> Alternative approach => Stacking (?)

4.4.10 Comparison to previous work

4.5 Execution Time Evaluation

Talk about constrained micro controller. Say that more frames potentially give more insight about fast gestures or that a longer battery life can be achieved. Say that we only consider COCD here, because that has the best feasible solution.

4.5.1 WCEP and WCET

4.5.2 AVR Compiler for AtMega328p

4.5.3 Optimization Level

Say that we considered O2 since the feasible solution can be compiled in O2. And why its a good Kompromiss between O3 and Os

4.5.4 Feature extraction

4.5.5 Tree Evaluation

4.5.6 Forest Evaluation

Here mainly talk about voting.

4.5.7 Total Execution Time

4.6 Size Evaluation

Explain that scikit learn provides some hyperparameters that can be changed. Elaborate which exist. Say that I considered here two parameters because all ensemble methods supported that. Maybe other reasons.

4.6.1 CCP (TODO: Abrev.)

4.6.2 Minimum Leaf Sample Size

Kapitel 5

Conclusion

Do a summary. Should be brief but also name everything and its result more or less. Say the improvement compared to the old approach. (Or downsides).

Mention future work. => Stacking

Literaturverzeichnis

[Ent20] ENTWICKLER scikit-learn: *1.10.6. Tree algorithms: ID3, C4.5, C5.0 and CART*. https://scikit-learn.org/stable/modules/tree.html#tree-algorithms. Version: 2020

[LR76] LAURENT, Hyafil ; RIVEST, Ronald L.: Constructing optimal binary decision trees is NP-complete. In: *Information processing letters* 5 (1976), Nr. 1, S. 15–17

[PVG+11] PEDREGOSA, F. ; VAROQUAUX, G. ; GRAMFORT, A. ; MICHEL, V. ; THIRION, B. ; GRISEL, O. ; BLONDEL, M. ; PRETTENHOFER, P. ; WEISS, R. ; DUBOURG, V. ; VANDERPLAS, J. ; PASSOS, A. ; COURNAPEAU, D. ; BRUCHER, M. ; PERROT, M. ; DUCHESNAY, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830

[Qui90] QUINLAN, J R.: Decision trees and decision-making. In: *IEEE Transactions on Systems, Man, and Cybernetics* 20 (1990), Nr. 2, S. 339–346

[SG14] SINGH, Sonia ; GUPTA, Priyanka: Comparative study ID3, cart and C4. 5 decision tree algorithm: a survey. In: *International Journal of Advanced Information Science and Technology (IJAIST)* 27 (2014), Nr. 27, S. 97–103

[Ste09] STEINBERG, Dan: Chapter 10 CART: Classification and Regression Trees. (2009), 01, S. 179–201

Anhang A

Content of the DVD

In this chapter, you should explain the content of your DVD.