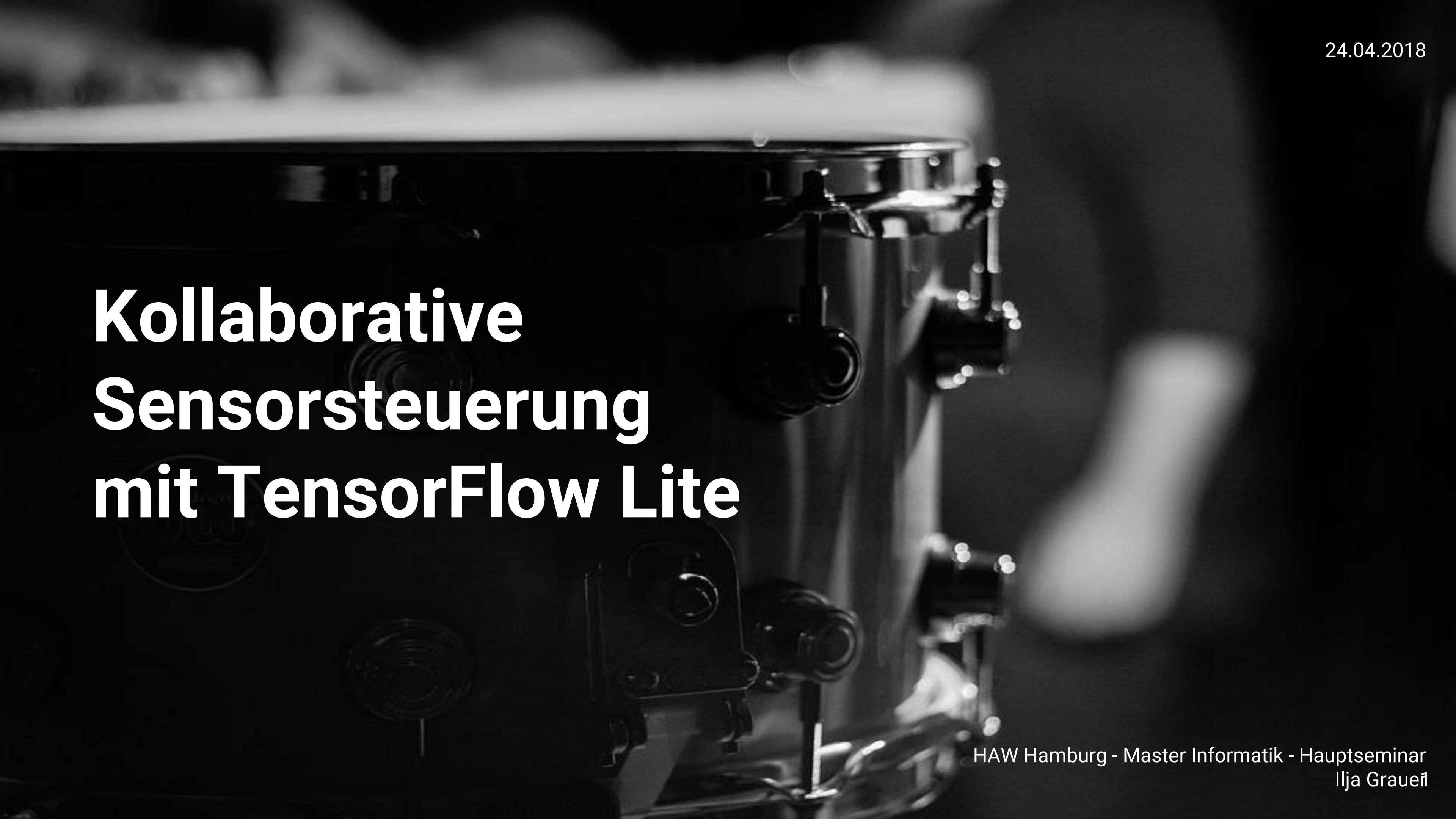


24.04.2018

# Kollaborative Sensorsteuerung mit TensorFlow Lite

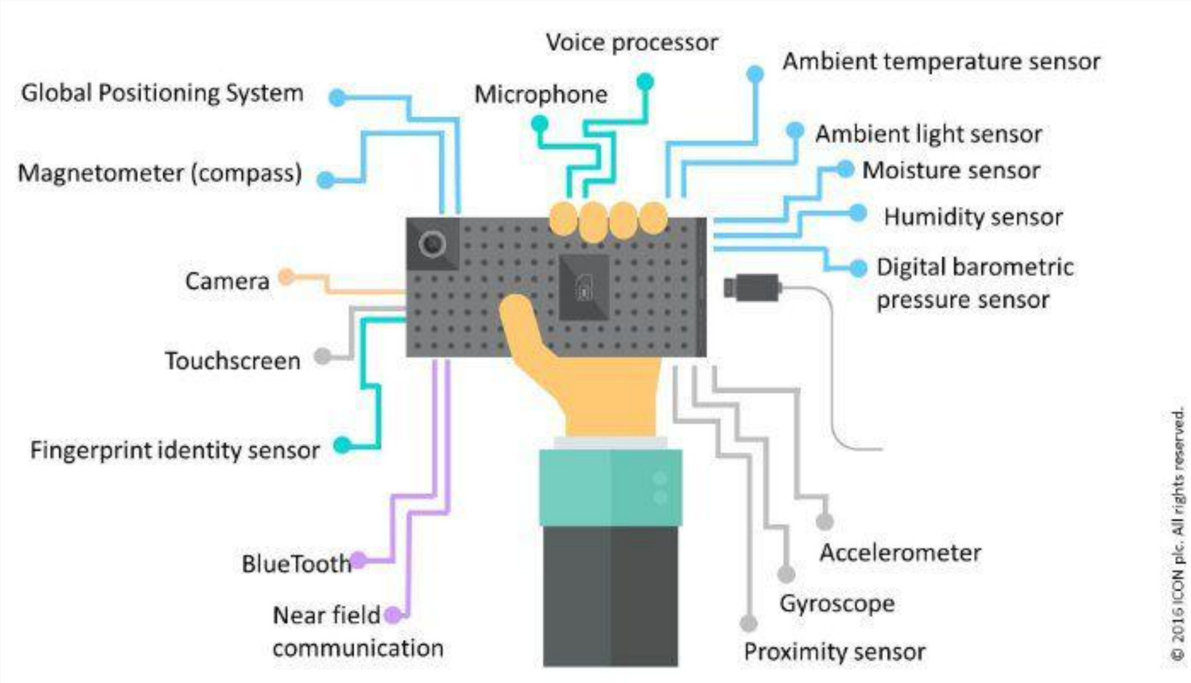
HAW Hamburg - Master Informatik - Hauptseminar
Ilja Grauel

# Agenda

- Motivation
- Problemstellung
- Grundlagen
  - Sensoren
  - TensorFlow Lite
  - Klassische Mobile Architektur
  - RNN/LSTM
- Stand der Forschung
  - (Erwähnung) ADTs
  - Remote Gestures
  - Microgestures
- Forschungsfrage
- Methodik
  - Big Picture
  - Datengenerierung
  - Clustering
  - Datenverarbeitung
  - TF Model
- Risiken

# Motivation

# Motivation - Sensoren



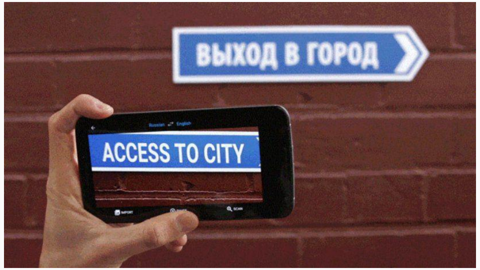

[B1]

- Viele Daten
- Mehr Rechenleistung
- Dedizierte Hardware (NPU)
- Optimierte KNNs (MobileNets, SqueezeNet)

↓

**Mobile Machine Learning**

# Motivation - Anwendungsbeispiele




[B2]


[B3]


[B4]

## Google Translate

- Bilderkennung
- Texterkennung
- ...

## Amazon Echo/Alexa

- Spracherkennung
- ...

## Smartwatch

- Pulserkennung
- ...

**Gemeinsamkeit: Erkennen externer Signale**

# Motivation - Steuerung


[1]

Vision: Schlagzeug-App mit Machine Learning

# Problem-stellung

# Problemstellung

## Problem 1: Bewegungen je Stick sehr identisch

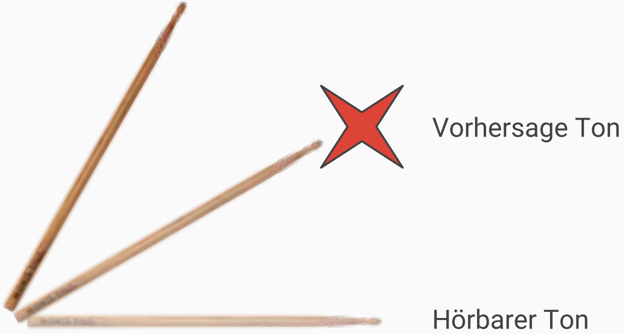
- Geräte könnten selben Ton ausspielen, egal welche Richtung
- Töne können sich überlagern

⇒ Geräte müssen miteinander kommunizieren


## Problem 2: Performance

- Töne könnten bei schnellem Spiel zu spät ausgespielt werden (QoS)

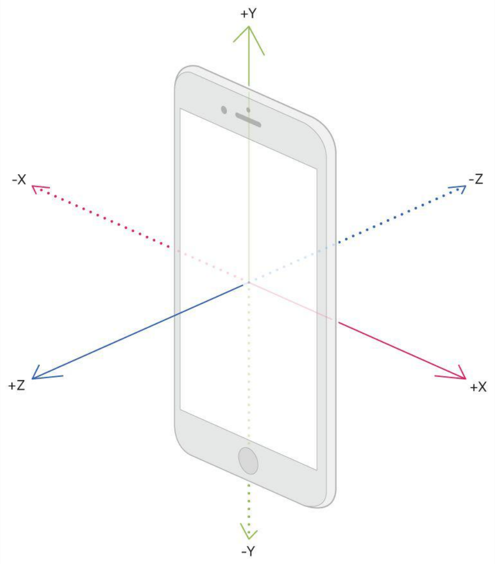
⇒ Vorzeitiges Ausspielen des Sounds (über Predictions)



# Grundlagen

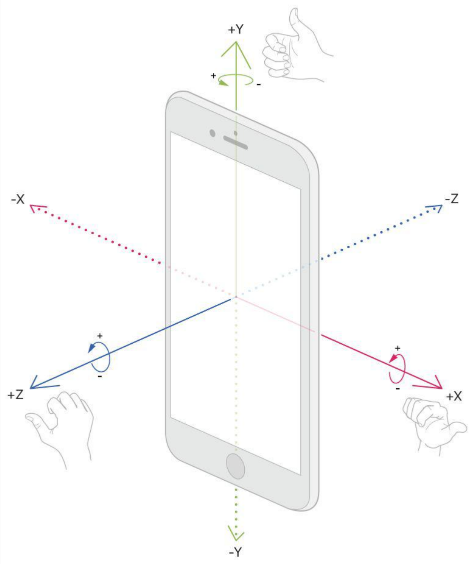
# Grundlagen - Sensordaten

## Accelerometer



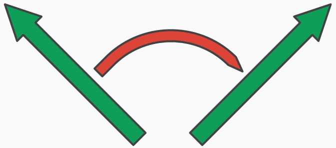

[B4]

## Gyroscope



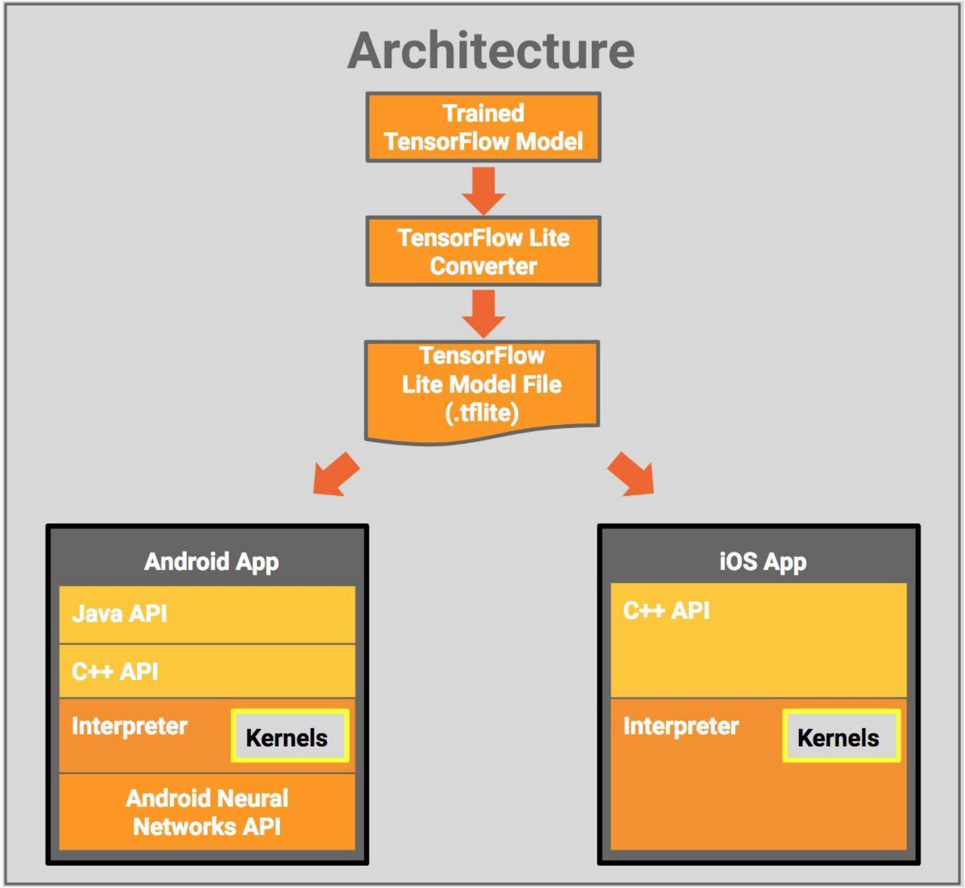

[B5]

## Magnetometer (Kompass)



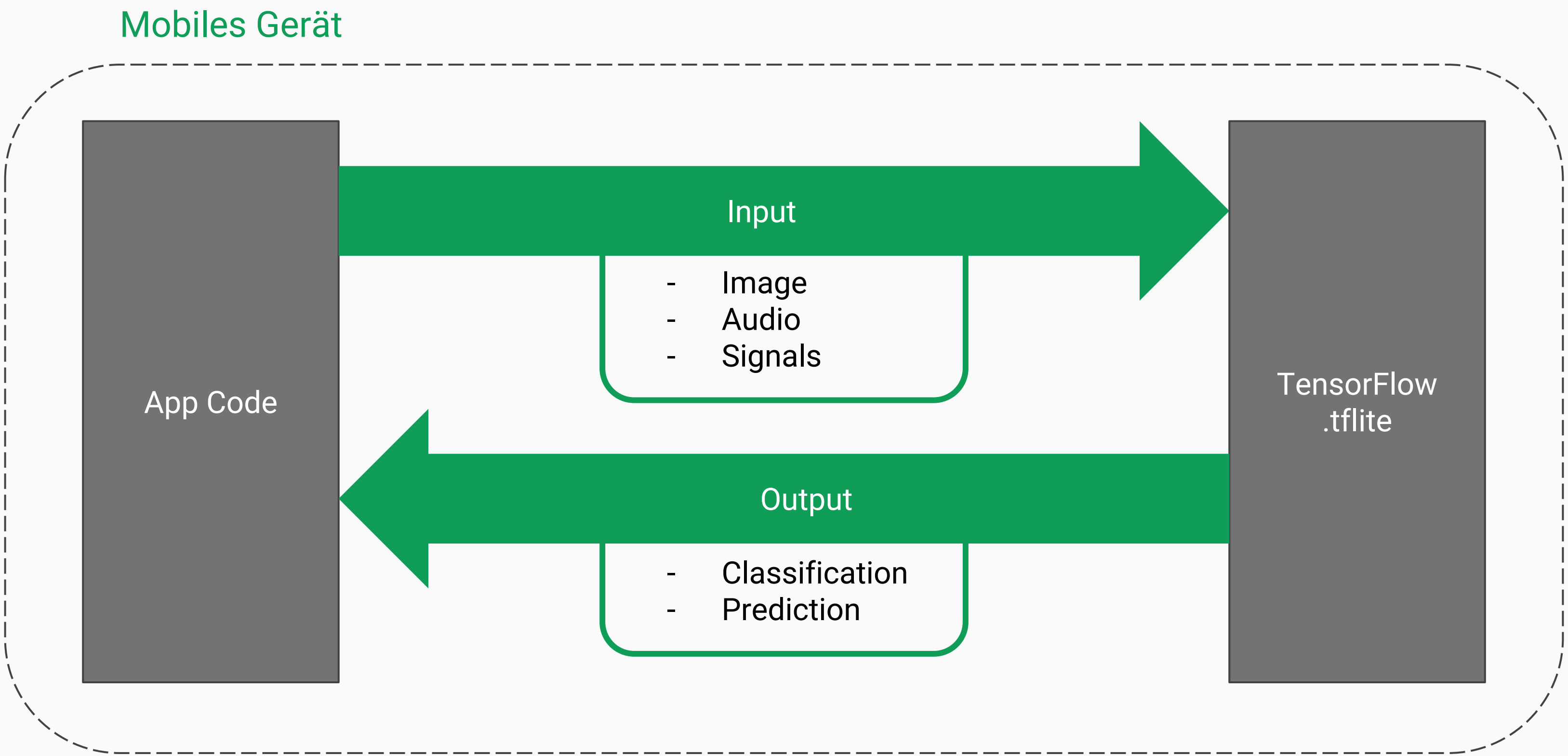
## Microphone

## Speaker


Architecture
Trained TensorFlow Model
TensorFlow Lite Converter
TensorFlow Lite Model File (.tflite)
Android App
Java API
C++ API
Interpreter
Kernels
Android Neural Networks API
iOS App
C++ API
Interpreter
Kernels

[6],[B5]

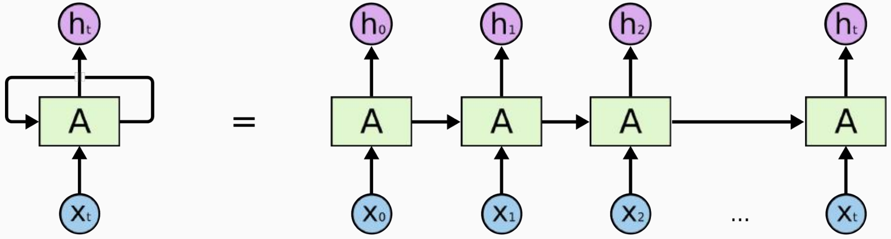
# Grundlagen - TensorFlow Lite

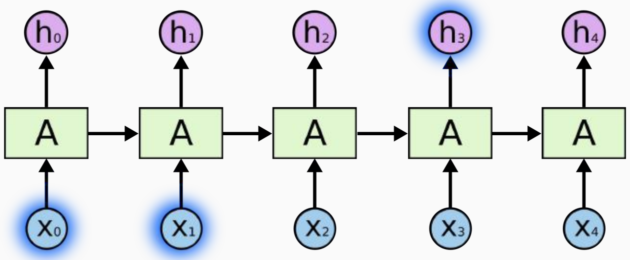


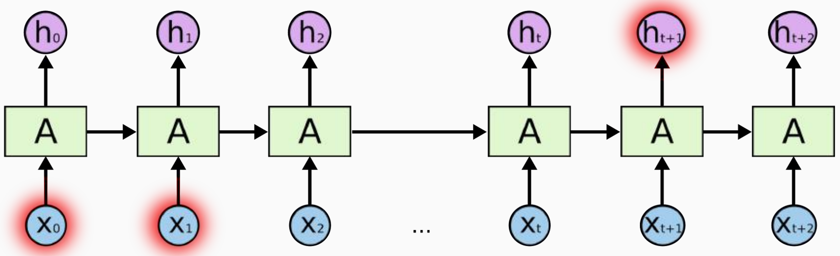
# Grundlagen - klassische mobile Architektur

# Grundlagen - RNN (Recursive Neural Network)

*RNN (entfaltet)*



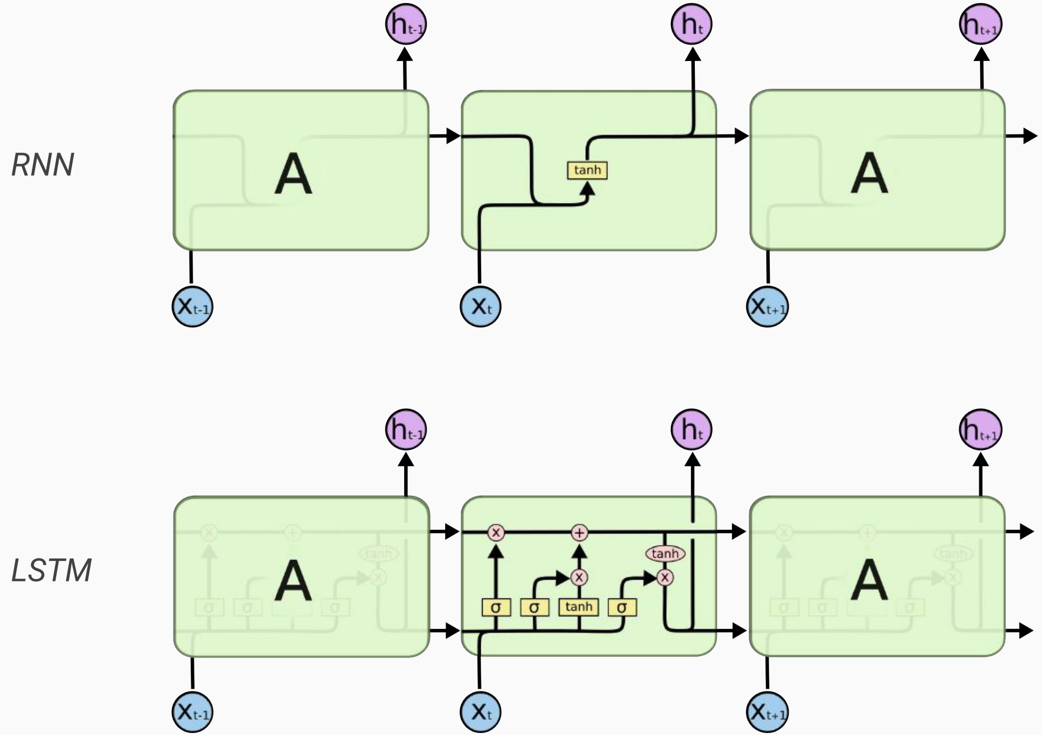




[14]

*RNN (short term memory)*

*RNN (long term memory) -> vanishing gradient problem*

# Grundlagen - LSTM (Long Short Term Memory)




[14]

# Stand der Forschung

# Forschungsstand - Automated Drum Transcription (Erwähnung)

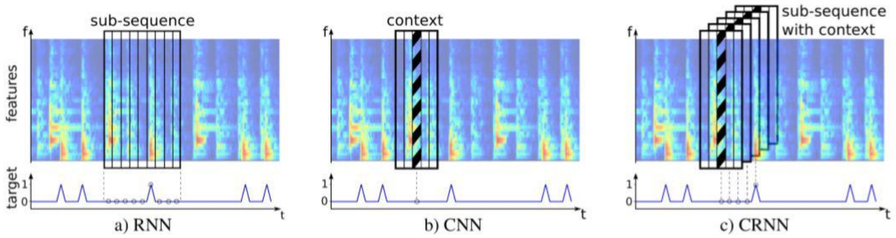



**Figure 2.** Comparison of mode of operation of RNNs, CNNs, and CRNNs on spectrograms of audio signals. RNNs process the input in a sequential manner. Usually, during training, only sub-sequences of the input signal are used to reduce the memory footprint of the networks. CNNs process the signal frame by frame without being aware of sequences. Because of this, a certain spectral context is added for each input frame. CRNNs, like RNNs, process the input sequentially, but additionally, a spectral context is added to every frame on which convolution is performed by the convolutional layers.

*Drum Transcription via Joint Beat and Drum. Modeling using Convolutional Recurrent Neural Networks [2]*

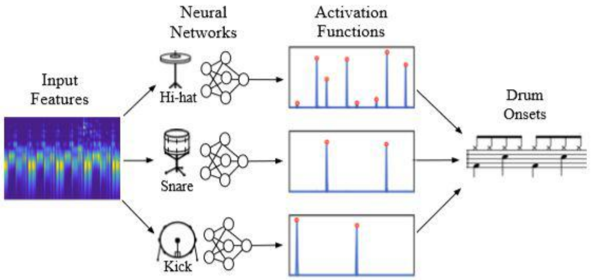



**Figure 1**: Overview of proposed method. Features are input to individual neural networks for each instrument, resulting in activation functions. Drum onsets are found by peak-picking the activation functions.

*Automatic Drum Transcription Using Bi-Directional Recurrent Neural Networks [3]*

Wie genau Trainingsdaten erzeugt werden sollen, ist zu dem Zeitpunkt noch unklar

# Forschungsstand - LSTM Gesture Remote Controller

*On the usage of smart devices to augment the user interaction with multimedia applications [4]*

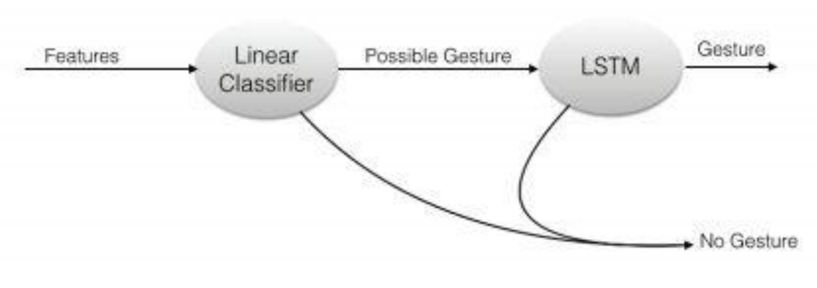
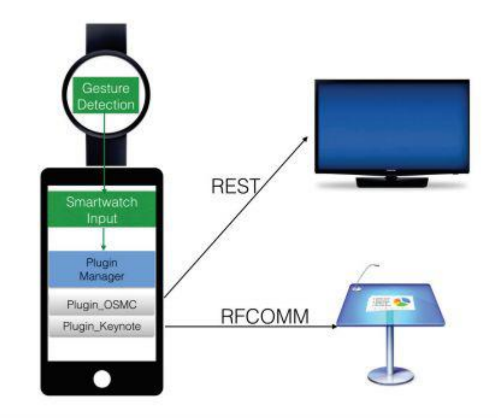



TABLE I: Activation Movements

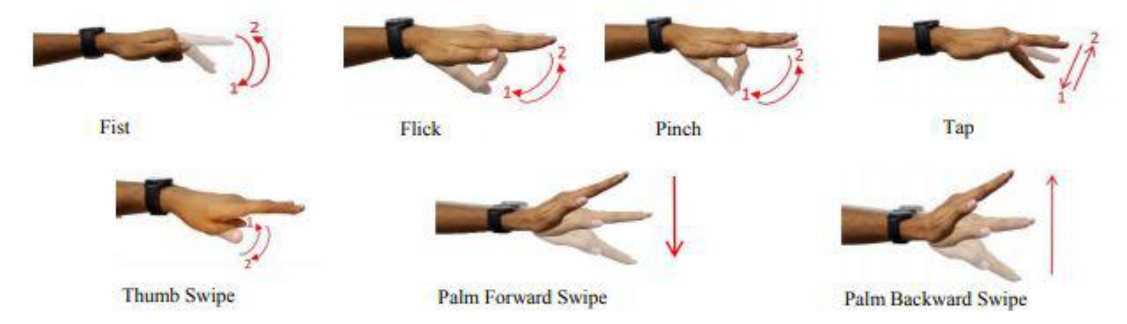
| Key | Movement | |
|---|---|---|
| L | Hand swipe to the left | |
| R | Hand swipe to the right | |
| F | Push forward gesture | |
| U | Arm and hand raised | |
| D | Arm and hand lowered | |
| C | Arm and hand rotated clockwise | |
| A | Arm and hand rotated anti clockwise | |

- Schlagzeug-Gesten sind sehr monoton → ggf. Andere Unterscheidung
- Kommunikation über Proxy (Smartphone) evtl zu langsam



# Forschungsstand - Microgestures

*Evaluation of Microgesture Recognition Using a Smartwatch [5]*



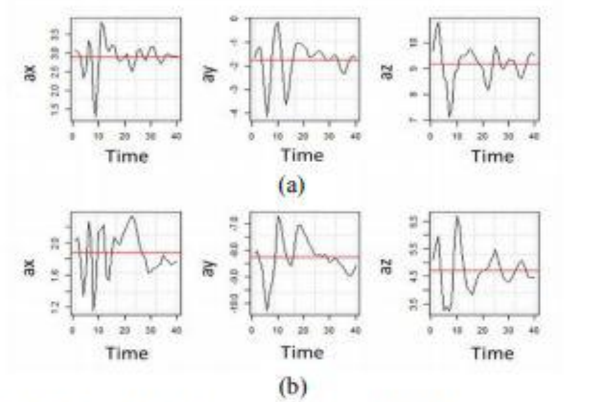




Fig 2. Accelerometer signals for Tap gesture when the palm is (a) horizontal. (b) tilted at 45°. The red line indicates the mean of the signal.

- SVM
- 7 Mikrogesten
- 94,4% bzw. 91,7% bei neuem Nutzer

⇒ kann helfen bei minimaler Lageverlagerung bzw Bewegungen der Handgelenke

# Forschungsstand - Zusammenfassung & Abgrenzung

1. Extraktion von Trainingsdaten
   => wie sollen qualitative Daten gewonnen werden?

2. Konnektivität Smartwatch und Smartphone über Android API
   ⇒ wie 2 Smartwatches (o.ä.) verbinden?

3. Gesten und Mikrogesten werden bereits gut erkannt
   ⇒ wie sehr ähnliche Schlagzeug-Gesten?

4. Wie sind die Delays?

# Forschungsfrage

# Forschungsfrage

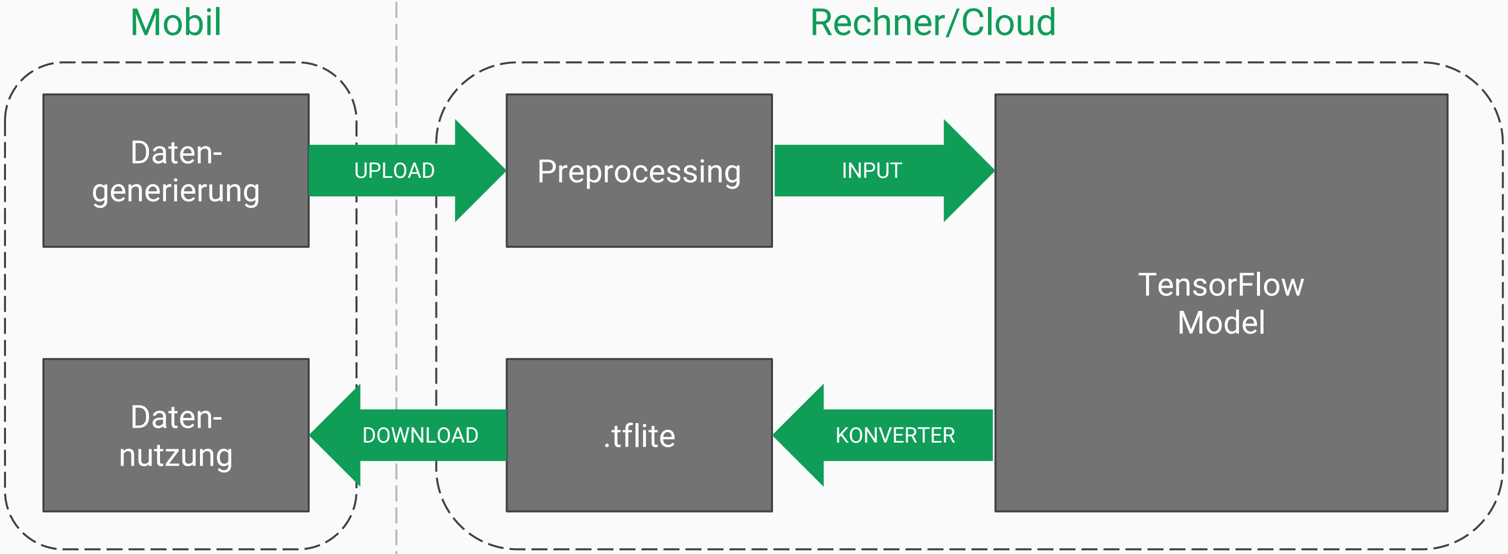
“Wie kann man mehrere sich örtlich nah beieinander befindende mobile Geräte mit jeweils eigenen ML-Modellen kollaborativ nutzen, um komplexe Gesten zur Steuerung von [Schlagzeug-Sounds] in Echtzeit zu ermöglichen?”

# Forschungsfrage - Rahmenbedingungen

| Teilfragen, Anforderungen | Definition, Metriken, Frameworks, Lösungsskizze |
|---|---|
| Geringe Distanz | Armlänge |
| Mobile Geräte | Smartwatches, Raspberry Pis |
| Embedded ML-Modelle | TensorFlow Lite |
| ML-Modelle dennoch verknüpft | Signalbrücke (NFC, Bluetooth, unhörbare Frequenz) |
| Gesten mit mehreren Geräten | Zwei Geräte |
| Steuerung | Output sind Sounds |
| Echtzeit | <=1s Verzögerung |

# Methodik

# Methodik - Big Picture



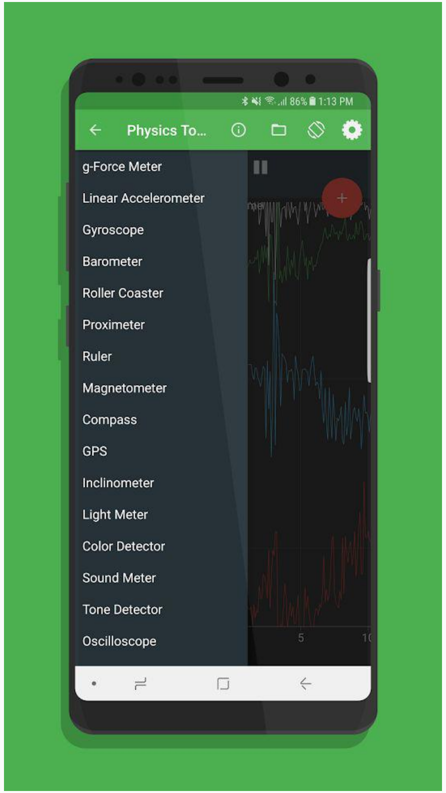

Trainingsdaten werden vom Nutzgerät generiert und genutzt

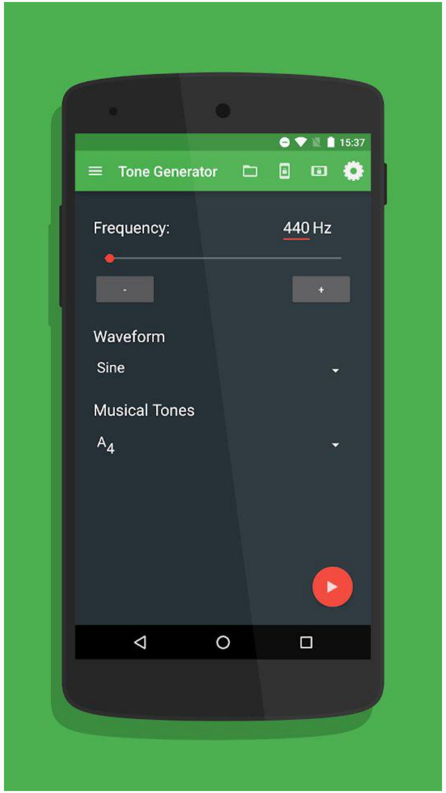
Daten werden aufbereitet, das Model wird trainiert und die .tflite exportiert

# Methodik - Datengenerierung

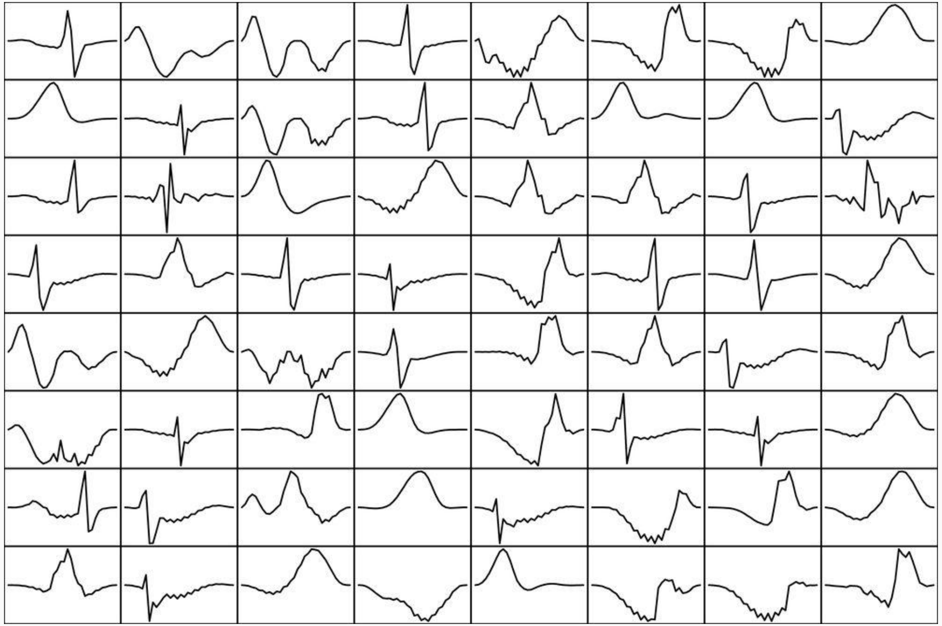
*Physics Toolbox Sensor Suite [13]*

App zur Ansicht und Speicherung von Sensordaten auf dem Smartphone

# Methodik - Clustering

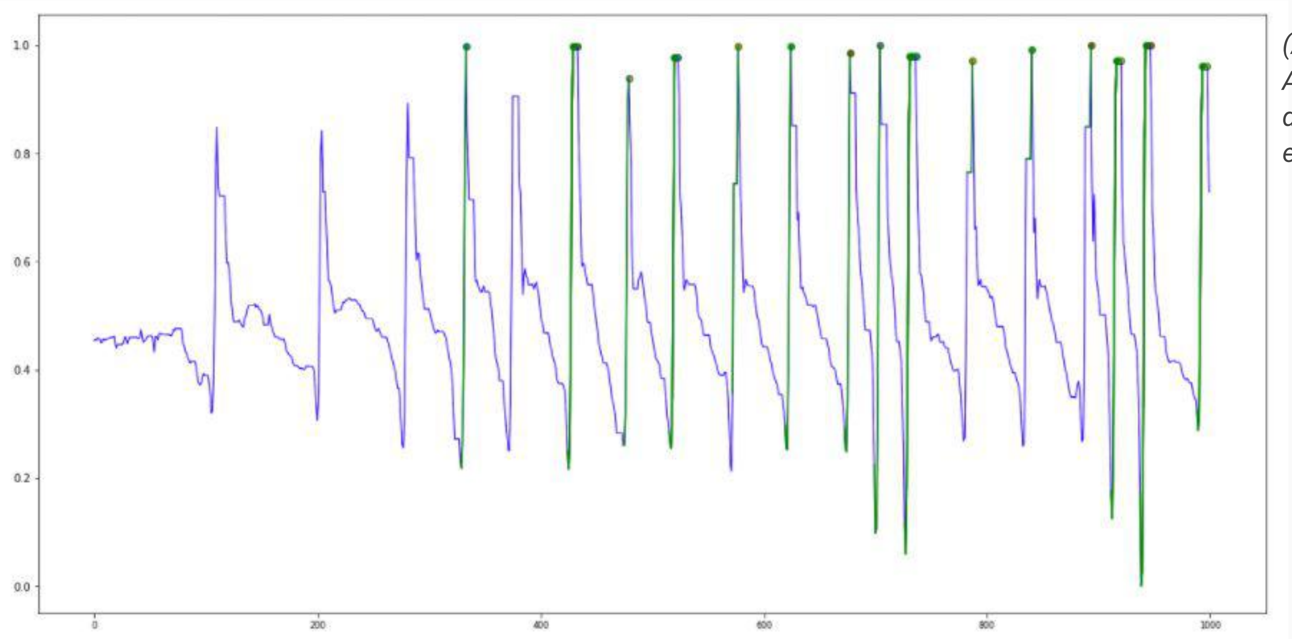


[B6]

Über Clustering-Verfahren sollen die unterschiedlichen Signale kategorisiert werden. Anschließend werden die Cluster gelabelt.

# Methodik - Preprocessing

## Peak Detection




*(Abbildung zeigt Accelerometer für die X-Achse, da es in dem Beispiel die einen eindeutigen Anschlag zeigt)*

Peak Detection läuft manuell und extrahiert eine Schlagbewegung (Höchster Ausholpunkt bis Schlagpunkt) aus allen Bewegungen

# Methodik - Preprocessing

Gravity Accelerometer

Sample 1

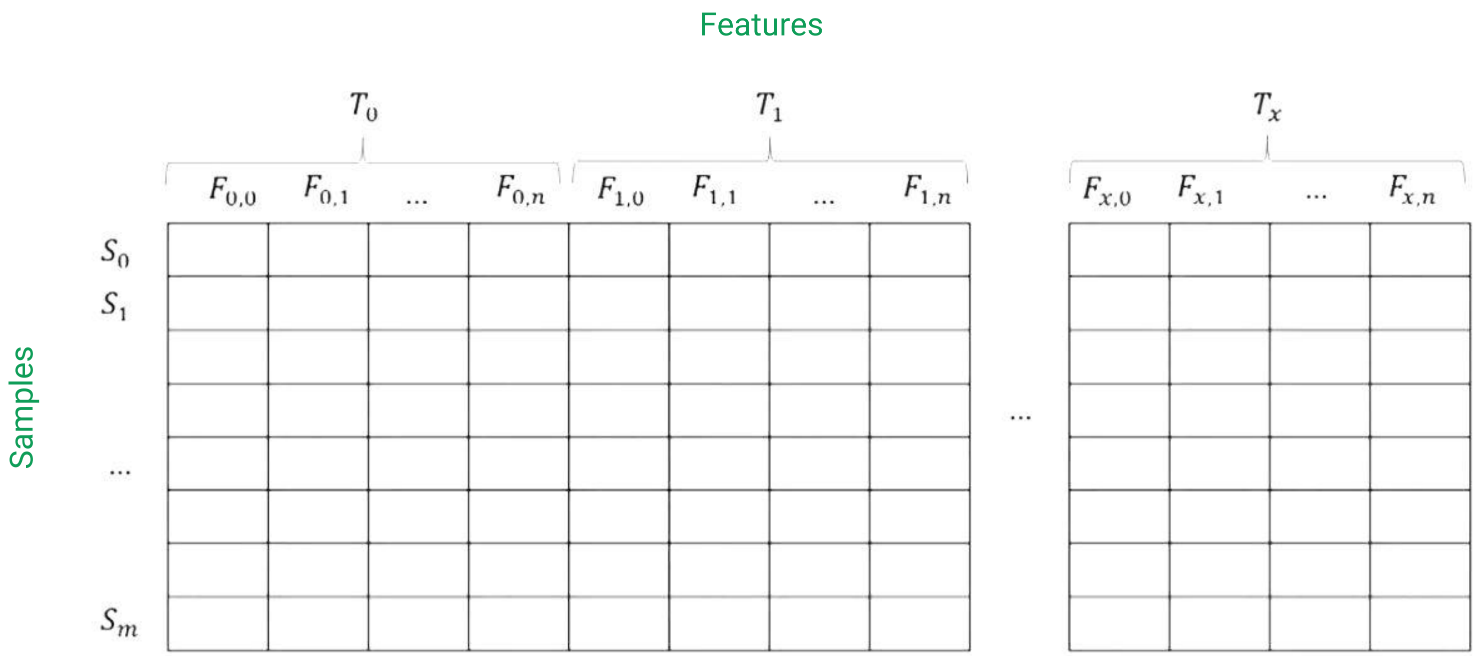
Sample 2

...

Sample n

| | gFx | gFy | gFz | ax | ay | az | wx | wy | wz | Bx | By | Bz | Azimuth | Pitch | Roll |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.453591 | 0.926698 | 0.212296 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.700970 | 0.000000 | 0.312125 |
| 1 | 0.455164 | 0.925443 | 0.215055 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.744012 | 0.010047 | 0.340448 |
| 2 | 0.454676 | 0.927079 | 0.213851 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.725967 | 0.006164 | 0.328061 |
| 3 | 0.458908 | 0.926883 | 0.210401 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.664087 | 0.022953 | 0.248991 |
| 4 | 0.456510 | 0.930402 | 0.208227 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.606477 | 0.011994 | 0.213085 |
| 5 | 0.456819 | 0.928355 | 0.209770 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.638732 | 0.013257 | 0.238946 |
| 6 | 0.450025 | 0.925721 | 0.222643 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.913835 | 0.020055 | 0.519223 |
| 7 | 0.449105 | 0.924650 | 0.228478 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.927266 | 0.036436 | 0.526447 |
| 8 | 0.455755 | 0.929106 | 0.221871 | 0.466503 | 0.756060 | 0.273829 | 0.501135 | 0.550665 | 0.385788 | 0.512078 | 0.021970 | 0.698080 | 0.818019 | 0.023587 | 0.400355 |
| 9 | 0.455755 | 0.929106 | 0.221871 | 0.466503 | 0.756060 | 0.273829 | 0.491927 | 0.553092 | 0.387788 | 0.512078 | 0.021970 | 0.698080 | 0.818019 | 0.023587 | 0.400355 |
| 10 | 0.455755 | 0.929106 | 0.221871 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.512078 | 0.021970 | 0.698080 | 0.818019 | 0.023587 | 0.400355 |
| 11 | 0.455755 | 0.929106 | 0.221871 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.814458 | 0.023587 | 0.400355 |
| 12 | 0.456668 | 0.923642 | 0.215827 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.739865 | 0.017542 | 0.329794 |
| 13 | 0.457039 | 0.921286 | 0.211068 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.661408 | 0.016115 | 0.259228 |
| 14 | 0.458358 | 0.924126 | 0.213652 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.704514 | 0.022477 | 0.289067 |
| 15 | 0.459492 | 0.923570 | 0.212284 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.689765 | 0.027096 | 0.269450 |
| 16 | 0.459725 | 0.925052 | 0.209922 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.659467 | 0.027294 | 0.243831 |
| 17 | 0.460048 | 0.923158 | 0.211173 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.678991 | 0.029559 | 0.256891 |
| 18 | 0.462219 | 0.920915 | 0.213430 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.718504 | 0.041298 | 0.273854 |
| 19 | 0.445127 | 0.922253 | 0.205502 | 0.468951 | 0.759718 | 0.266396 | 0.491927 | 0.553092 | 0.387788 | 0.509068 | 0.026698 | 0.682288 | 0.154830 | 0.017536 | 0.823075 |

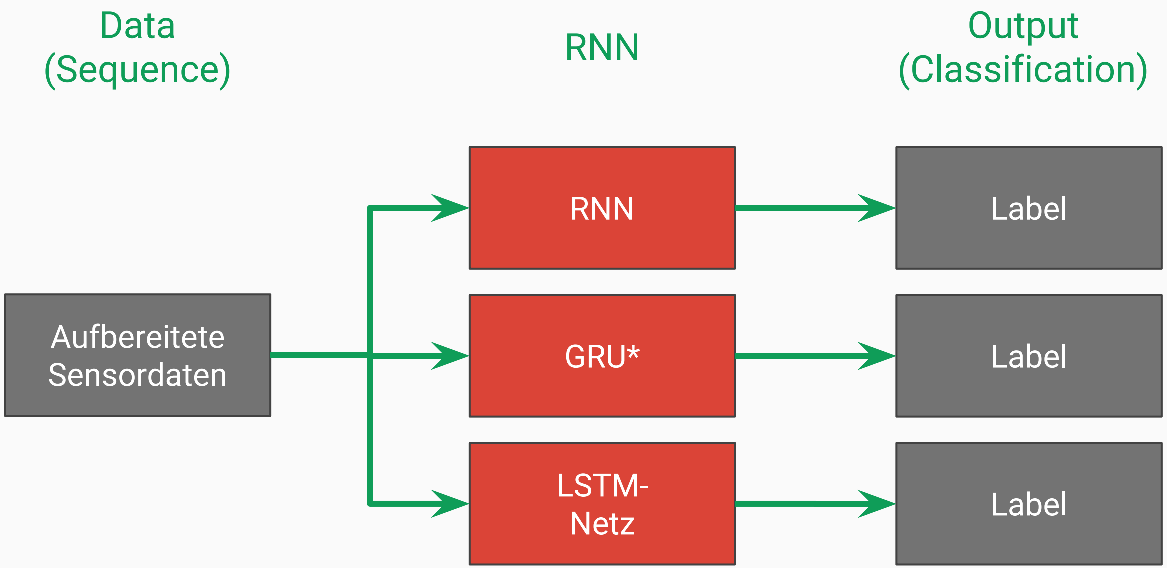
# Methodik - Preprocessing




S = samples
F = features
T = time steps

m = # samples
n = # features
x = # time steps

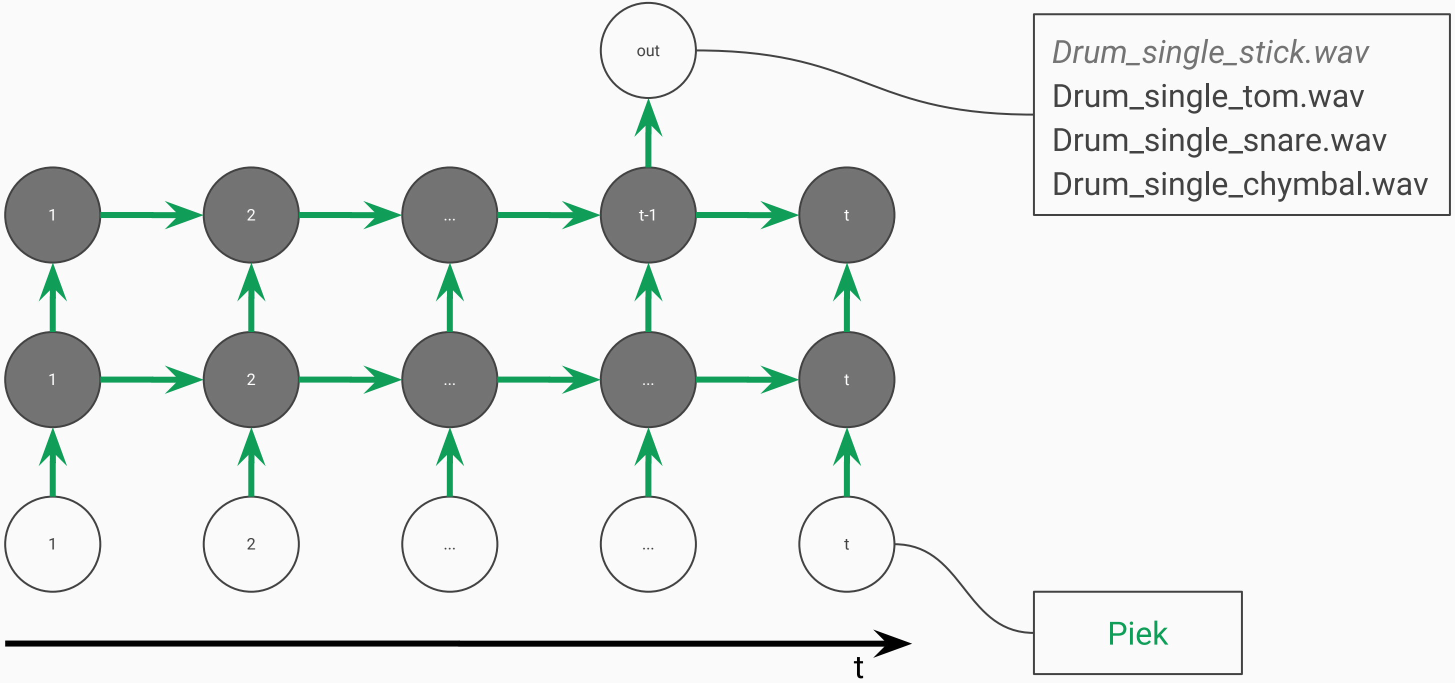
# Methodik - TensorFlow Model




Im Grundprojekt soll evaluiert werden, welches Netz die beste Performance bringt

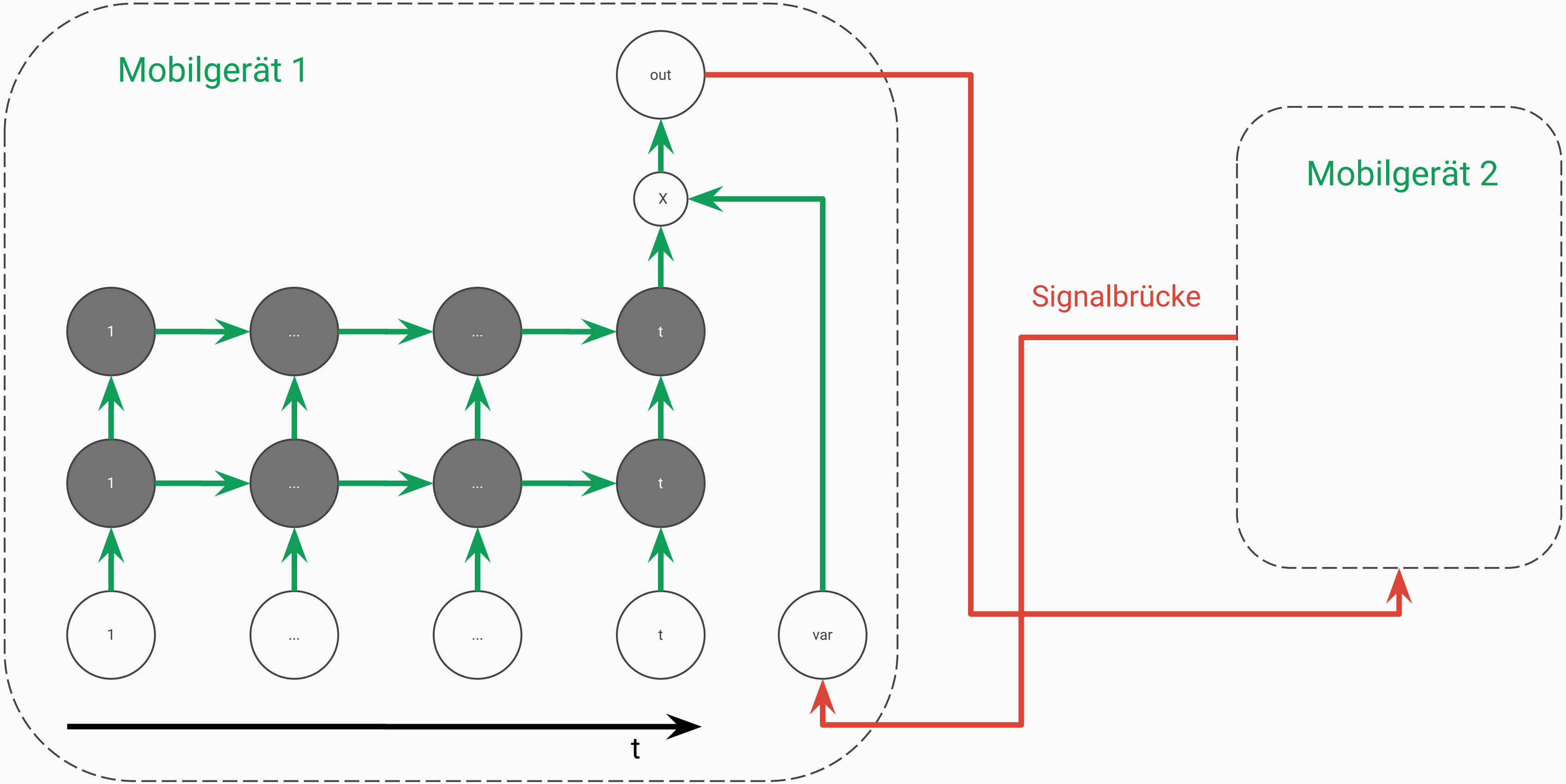
**GRU → Gated Recurrent Network,* Cho, et al., *2014)*

# Methodik - TensorFlow Model: Predicted Classification




Der Aufschlag soll vorhergesagt werden, um eventuelle Delays zu vermeiden

# Methodik - TensorFlow Model: Kollaboration (Konzept)



# Risiken

# Risiken

- ❏ Unzureichende Trainingsdaten in Quantität und Qualität
  Lösung ⇒ ADT, Unsupervised Clustering
- ❏ Performance / Quality Of Service (Latenz, Überlagerung)
  Lösung ⇒ Predicting
- ❏ Hardware-Konstruktion
  Lösung => Kalibrierung
- ❏ Overfitting
  Lösung ⇒ Early Stopping, Dropout
- ❏ Debugging
  Lösung => TensorBoard

# Danke!
# Fragen?

# Quellen


[1] https://www.freedrum.rocks/

[2] Richard Vogl, Peter Knees et. al. *Drum Transcription via Joint Beat and Drum. Modeling using Convolutional Recurrent Neural Networks.* Proceedings of the 18th ISMIR Conference, Suzhou, China, October 23-27, 2017

[3] Carl Southall, Ryan Stables, Jason Hockman. *Automatic Drum Transcription Using Bi-Directional Recurrent Neural Networks*. Proceedings of the 17th ISMIR Conference, New York City, USA, August 7-11, 2016

[4] A. Ferrari, V. Galli, D. Puccinelli and S. Giordano. *On the usage of smart devices to augment the user interaction with multimedia applications*. 2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Macau, 2017, pp. 1-9.

[5] S. Agarwal and S. Ghosh. *Evaluation of Microgesture Recognition Using a Smartwatch*. 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, 2017, pp. 986-991

[6] https://www.tensorflow.org/

[7] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, 2017

[8] https://developer.android.com/guide/topics/sensors/sensors_overview.html

[9] https://propakistani.pk/2017/07/25/heres-brief-intro-fancy-sensors-smartphone/

[10] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11] https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714

[12] http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[13] https://play.google.com/store/apps/details?id=com.chrystianvieyra.physicstoolboxsuite

# Bildquellen


[B1] http://www.servicesmobiles.fr/wp-content/uploads/2017/09/sensor_smartphone.jpg

[B2] https://img.zeit.de/digital/internet/2015-01/google-translate/wide__820x461__desktop

[B3] http://picscdn.redblue.de/doi/pixelboxx-mss-75879035/fee_786_587_png/AMAZON-Echo-Dot-2.-Generation-Smart-Speaker-mit-Sprachsteuerung--Schwarz

[B4] https://shop.telekom.de/system/images/attachments/000/024/054/original/1497969414/fenix-5s-gps-multisport-smartwatch-schwarz-silber-list.png?1497969414

[B5] https://www.tensorflow.org/images/tflite-architecture.jpg

[B6] https://www.safaribooksonline.com/library/view/practical-machine-learning/9781491914151/images/pmad_0403.png