

# Implementierung und Evaluation von Deep Learning Architekturen für die Klassifikation von Videos

von Yannick Kloss

# Agenda

1. Der Datensatz
2. CNN basierte Verfahren
3. Zu untersuchende Architekturen
4. Ergebnisse
5. Nachtrag

# Der Datensatz

HMDB51 - Human Motion Database<sup>1</sup>:

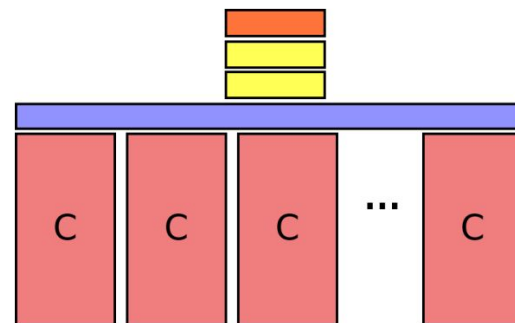
- 51 Kategorien
- 6849 Video Clips
- > 100 Clips pro Kategorie
- Die besten Algorithmen:
  - Improved Dense Trajectory: 57,2%
  - W-Flow Dense Trajectories: 52,1%
  - Dense Trajectories: 46,6%

<sup>1</sup>HMDB51: <http://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/#dataset>

# CNN basierte Verfahren

## Feature-Pooling Architekturen<sup>1</sup>:

1. CNN extrahiert Features einzelner Frames
2. Max-pooling über alle Features
3. Klassifizierung



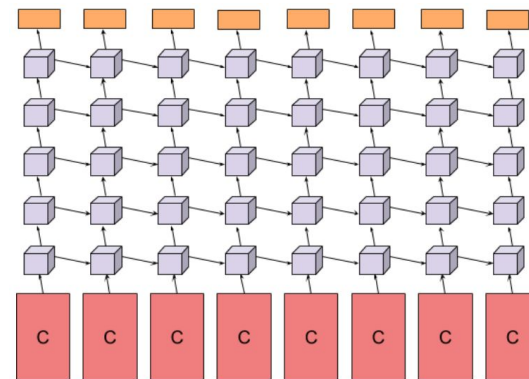
Conv Pooling<sup>1</sup>: Convolution (rot),  
max-pooling (blau), fully connected (gelb)  
und softmax (orange).

<sup>1</sup>J. Ng et al. (2015): Beyond Short Snippets: Deep Networks for Video Classification. In ArXiv e-prints, 1503.08909.

# CNN basierte Verfahren

## LSTM Architekturen<sup>1</sup>:

1. CNN extrahiert Features einzelner Frames
2. Features als Input für LSTM Schichten
3. Klassifizierung



Deep Video LSTM<sup>1</sup>: Convolution (rot),  
LSTM Zellen (blau) und softmax (orange).

<sup>1</sup>J. Ng et al. (2015): Beyond Short Snippets: Deep Networks for Video Classification. In ArXiv e-prints, 1503.08909.

# Zu untersuchende Architekturen

⇒ Feature-Pooling und LSTM Architekturen auf Basis einer CNN Architektur implementieren und evaluieren.

## **CNN-Feature-Extraction:**

- Annahme: Trainieren von komplexen CNN Architekturen ist aufwendig
- Idee: Mit *Transfer Learning* eine vor-trainierte CNN Architektur umtrainieren

# Zu untersuchende Architekturen

## Inception-v3 umtrainieren:

- Mit extrahierten Frames der Videos die letzten beiden Inception-Module und die Klassifizierungsschichten trainieren

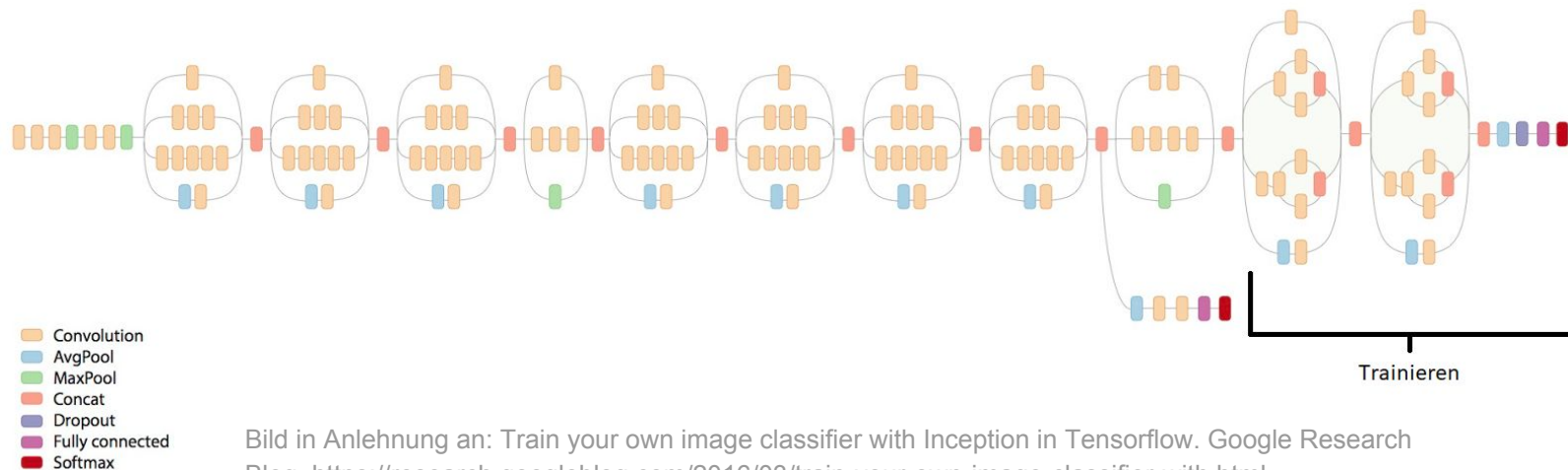
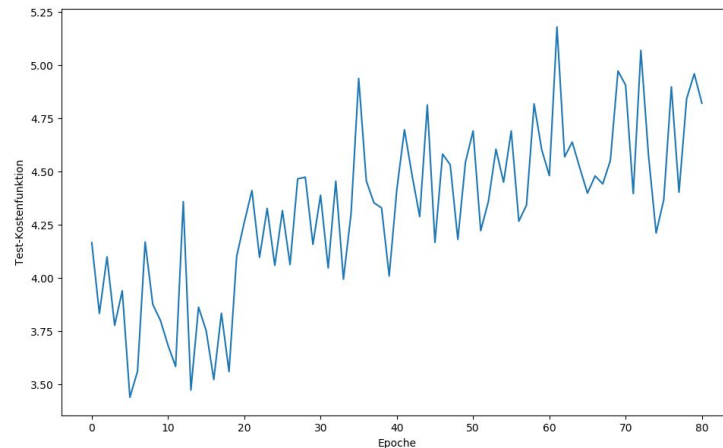
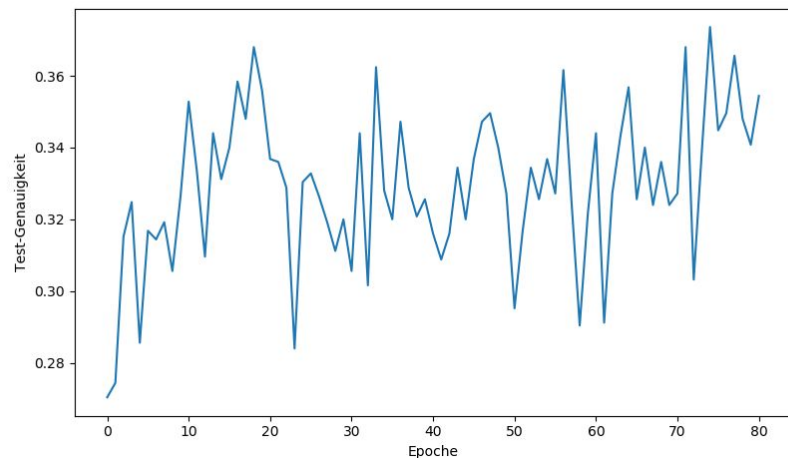


Bild in Anlehnung an: Train your own image classifier with Inception in Tensorflow. Google Research Blog, <https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html>.

# Zu untersuchende Architekturen

Inception-v3 umtrainieren:

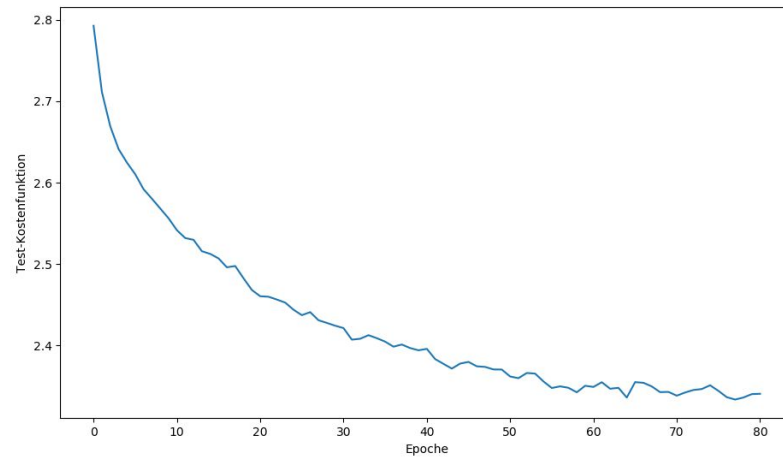
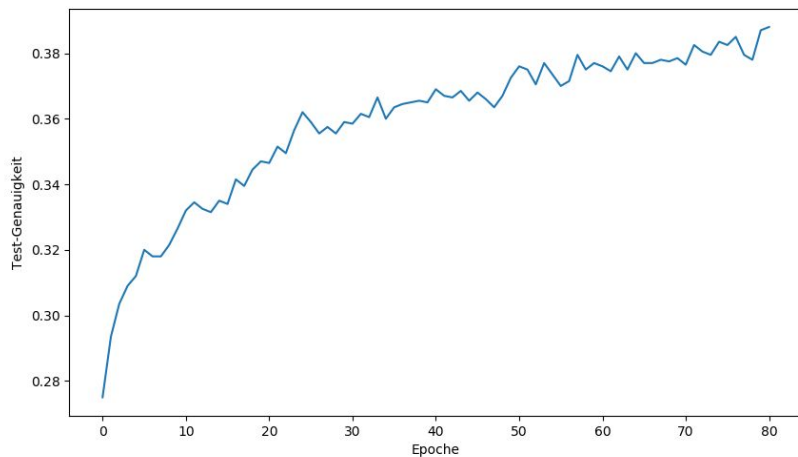


➡ **Overfitting** nach wenigen Epochen



# Zu untersuchende Architekturen

Inception-v3 umtrainieren (mit Dropout):



# Zu untersuchende Architekturen

Features extrahieren:

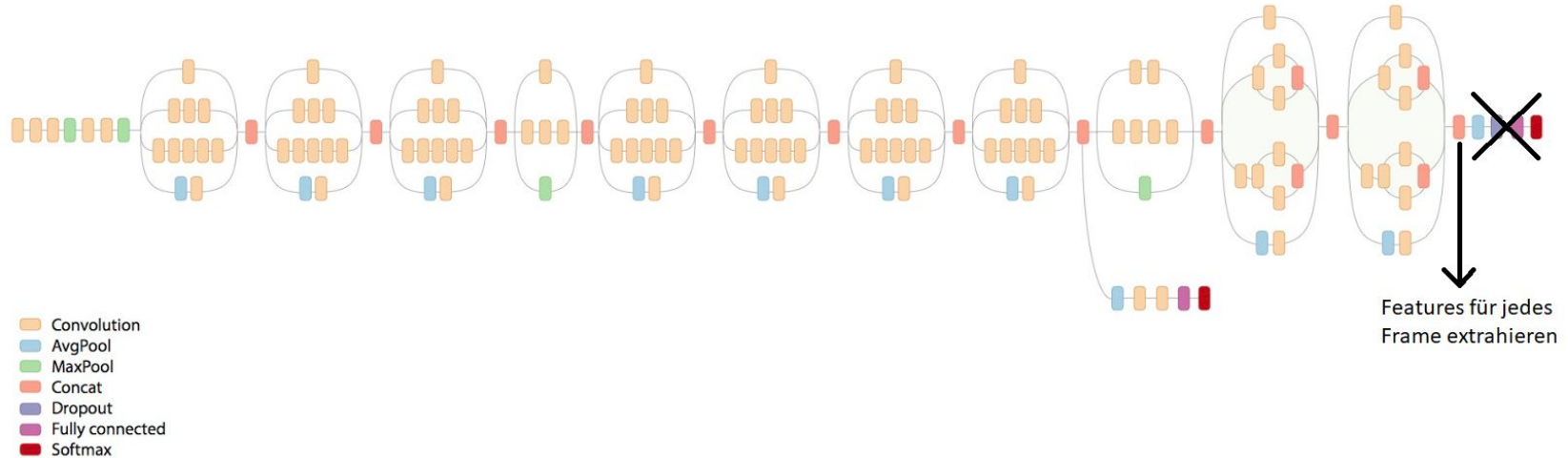


Bild in Anlehnung an: Train your own image classifier with Inception in Tensorflow. Google Research Blog, <https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html>.

# Zu untersuchende Architekturen

## Bezüglich Feature-Pooling Architektur:

- *Conv Pooling* hat höchste Genauigkeit
- Max-pooling Schicht: Was genau wird gepoolt?

# Zu untersuchende Architekturen

## Zu untersuchen:

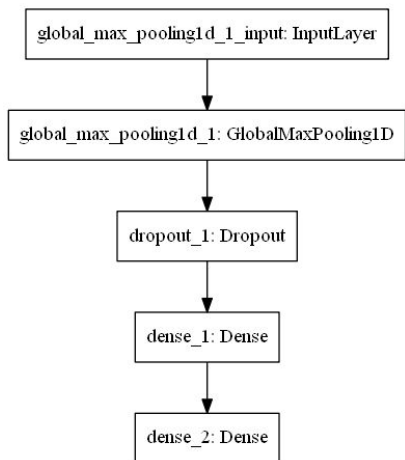
- Vor-trainierte CNN Architektur verwenden oder umtrainieren?
- Feature-Pooling Architektur: *Globales* 1D-max-pooling oder 1D-max-pooling?
- Klassifizierung: Schichten Topologien untersuchen

# Ergebnisse

Einschränkungen:

- Fast keine Vorverarbeitung
- Nur 40 Frames pro Video
- Kein Hyperparameter tuning

# Ergebnisse



## Feature Pooling mit trainierten Inception-v3

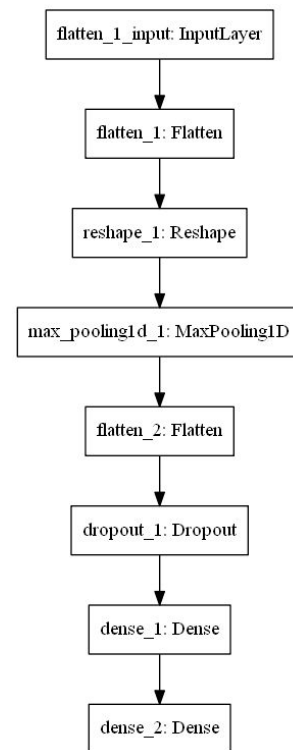
Global 1D-max-pooling	6,06%
-----------------------	-------

1D-max-pooling	6,35%
----------------	-------

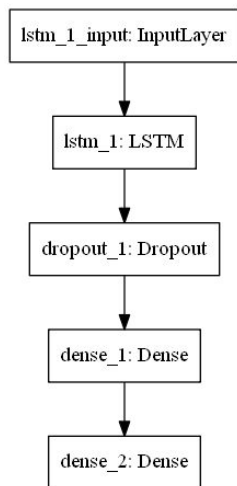
## Feature Pooling mit Inception-v3

Global 1D-max-pooling	<b>43,32%</b>
-----------------------	---------------

1D-max-pooling	<b>39,86%</b>
----------------	---------------



# Ergebnisse



<b>LSTM mit trainierten Inception-v3</b>	5,27%
<b>LSTM mit Inception-v3</b>	38,27%

# Nachtrag

Möglichkeiten für das Verbessern Genauigkeit:

- Hyperparameter tuning
- Bildvorverarbeitung
- Mit allen Frames trainieren