Praktikum Mobile und Verteilte Systeme

Machine Learning for Mobile Platforms

Prof. Dr. Claudia Linnhoff-Popien
André Ebert, Sebastian Feld, Thomy Phan
http://www.mobile.ifi.lmu.de

SoSe 2018
→ Machine Learning
What is Machine Learning?

• **Goal:** Create programs that learn how to solve complex problems

• Learn statistical models from experience / data

• Use learned models for e.g.
  – Object Recognition
  – Prediction
  – Control
  – Compression
  – Data Generation
Why Machine Learning?

- **Goal:** Create programs that learn how to solve complex problems

- Many problems cannot be solved by engineering hard-coded solutions
  - Too many aspects to consider
  - Too many rules
  - Hard adaption to changes
  ...

- Examples:
  - Object recognition in images
  - Natural language processing
  - User behaviour analytics
  - Locomotion
Types of Machine Learning

Unsupervised Learning

Supervised Learning

Reinforcement Learning
Types of Machine Learning

- Unsupervised Learning
- Supervised Learning
- Reinforcement Learning
Supervised Learning

- **Goal**: Learn an unknown function $f: X \rightarrow Y$ from labeled data
- Data consists of input-output pairs $(x, y)$ with $x \in X, y \in Y$
- Approximate $f$ by learning a general mapping between $(x, y)$
- Examples:

  ![Classification](image1)
  ![Regression](image2)

**Classification**

**Regression**
Learning Approaches

Bayes Classification

Linear Classification & SVMs

Decision Trees & Forests

(Deep) Neural Networks
Supervised Learning Pipeline

- **Training Data**
- **Test Data**
- **New Data**

- **Training**
- **Validation**
- **Application**
Challenges of Machine Learning

• Data Availability
• Data Complexity
• Efficiency
• Compactness
• Interpretability
• Robustness
• Adaptivity
→ Machine Learning for Mobile Platforms
Applications

- Personalized Content
- Recommender Systems
- Realtime Prediction
- Fraud Detection
- Augmented Reality
Architectures

Training

Prediction on Server

Prediction on Device

Model

Model

Model
Prediction on Mobile Devices (1)

- Improved User Experience
  - Latency
  - Availability
  - Speed
  - Privacy

- Lower development and operation costs
  - No server farm required
  - No client-server interaction
Prediction on Mobile Devices (2)

- **Model inside** the App
  - Easy to deploy
  - Model hidden

- **Model alongside** the App
  - Easy to update model
  - Smaller binary
→ Machine Learning on Android
Neural Network API

- New since Android 8.1
- Part of the NDK (C API)
- base layer for higher-level machine learning frameworks (e.g. TensorFlow)

 TensorFlow

- Framework for high performance numerical computation
- Originally developed by Google Brain
- Open-source under Apache 2.0 License
- Deep learning support
- Runs on different platforms (CPU, GPU, TPU, etc.)
TensorFlow API

- Python, C++, Java
- Tensors as data structure (N-dimensional arrays)
- TensorFlow operations work with Tensors
- Computations can be expressed as Graph

**Example:** Deep Convolutional Neural Network

Source: https://www.tensorflow.org/programmers_guide/graphs
TensorFlow Example (1)

• Building the Computational Graph

```python
import tensorflow as tf

# Neural network architecture
input = tf.placeholder(shape=[batch_size, h, w, c], dtype=tf.float32)
h1 = tf.layers.conv2d(input, 32, 5, 1, tf.nn.relu)
h2 = tf.layers.max_pooling2d(h1, 2, 1)
h3 = tf.layers.conv2d(h2, 32, 3, 1, tf.nn.relu)
h4 = tf.layers.max_pooling2d(h3, 2, 1)
h5 = tf.layers.flatten(h4)
h6 = tf.layers.dense(h5, 256, tf.nn.relu)
h7 = tf.layers.dropout(h6)
h8 = tf.layers.dense(h7, nr_outputs, tf.nn.relu)

# Training operation
labels = tf.placeholder(shape=[minibatch_size, nr_outputs], dtype=tf.float32)
loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits_v2(labels=labels, logits=h8))
train_op = tf.train.AdamOptimizer().minimize(loss)

# Prediction operation
P = tf.softmax(h8)
```
TensorFlow Example (2)

- Possible Operations

```python
# Training operation
label = tf.placeholder(shape=[minibatch_size, nr_outputs], dtype=tf.float32)
loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits_v2(labels=label, logits=h8))
train_op = tf.train.AdamOptimizer().minimize(loss)

# Prediction operation
P = tf.softmax(h8)
```

- Running the Computational Graph

```python
init_op = tf.global_variables_initializer()
session = tf.Session()
session.run(init_op)

# Training the neural network
session.run(train_op, feed_dict={input:inputs, label:labels})

# Prediction with the neural network
prediction = session.run(P, feed_dict={input:inputs})
session.close()
```
TensorFlow Lite

- Lightweight version for mobile/embedded devices
- Supports hardware acceleration (Neural Network API)
- Optimization of models for mobile apps
- Runs on Android and iOS
- Documentation on [https://www.tensorflow.org/mobile/tflite/](https://www.tensorflow.org/mobile/tflite/)
TensorFlow Lite Architecture

Source: https://www.tensorflow.org/mobile/tflite/
TensorFlow Lite – Pre-tested Models

Pre-tested models available:


Models can be retrained for custom purposes
TensorFlow - Next Steps

For the curious ones:
• Go to https://www.tensorflow.org/mobile/tflite/
• Install the Demo App and get familiar with the code

For the ambitious ones:
• Get familiar with the TensorFlow API
  https://www.tensorflow.org/programmers_guide/
• Build and train models from scratch
• Retrain existing models
• Deploy them on an App (e.g. the Demo App)