Fallstudien II

Laura Kampmann, Christian Peters, Alina Stammen

12. Dezember 2020

Inhalt

Einleitung

Einleitung

- **Motivation**: Verbesserung mobiler Kommunikation von Endgäten
  $\rightarrow$ Vermeiden von z.B. "packet loss" und als Folge auch Retransmission
- Wie kann das erreicht werden?
  $\rightarrow$ Datenratenprädiktion um optimalen Zeitpunkt zum Senden von Daten zu ermitteln

Datenbeschreibung

Situation:

- echt Welt Messungen im öffentlichen LTE Netzwerk der 3 deutschen Mobilfunkanbieter o2, T-Mobile und Vodafone
- Aufteilung in mehrere Szenarien: "campus", "urban", "suburban" und "highway"
- pro Mobilfunkanbieter und Szenario wurden 10 Testfahrten durchgeführt

Datenbeschreibung

- "context": passive Messungen 1s
  → **RSRP**
  → **RSRQ**
  → **CQI**
  → **TA**
  → **velocity**
  → **Cell ID**
  → **payload size**
- "ul" / "dl": aktive Messungen 10s
  → **throughput** - Datenrate
- "cells": → RSRP / RSRQ der Nachbarzellen

Task I: Data Rate Prediction

Task I: Data Rate Prediction

- Ziel: Evaluation von neuen *anticipatory vehicular communication systems* durch möglichst realitätsnahe Simulationen [3]

  $\Rightarrow$ Ansatz: *Data-Driven Network Simulation*

- Durch Machine Learning Modelle sollen möglichst realistische Vorhersagen der Datenraten generiert werden
- Hoffnung: Bessere Aussagekraft der Simulationen durch Einsatz echten Datenmaterials

Task I: Data Rate Prediction

Gradient Boosted Trees

Gradient Boosted Trees

- Kann man aus vielen "schwachen" Lernern einen starken Lerner konstruieren?
  - $\Rightarrow$ Ja, Boosting ist eines der mächtigsten Konzepte des Machine Learning [2]
- Kombination von einfachen CART Bäumen zu einem starken Ensemble
  - $\Rightarrow$ Ähnlich zu Random Forest
- Der Unterschied zum Random Forest liegt im Training!

Training von Gradient Boosted Trees

- Bäume werden nacheinander zum Ensemble hinzugefügt
- Jeder neue Baum versucht, die Schwächen seiner Vorgänger "auszubügeln"

  $\Rightarrow$ *Additives Training*
- Je mehr Bäume aufgenommen werden, desto geringer wird der Training-Error (das Modell wird aber komplexer)

  $\Rightarrow$ Kontrolle des *Bias-Variance Tradeoffs*

  $\Rightarrow$ Zusätzlich gibt es Regularisierungs-Parameter

Implementierung: XGBoost

- Liefert state-of-the-art Performance in einer Vielzahl von ML-Problemen
- In 2015 haben 19/25 Gewinner von Kaggle-Competitions XGBoost eingesetzt
- Kann problemlos auf mehrere Milliarden Training Samples skaliert werden
- Lässt sich aber auch hervorragend auf ressourcenbegrenzten Systemen einsetzen [1]

Task I: Data Rate Prediction

ARIMA

Situation

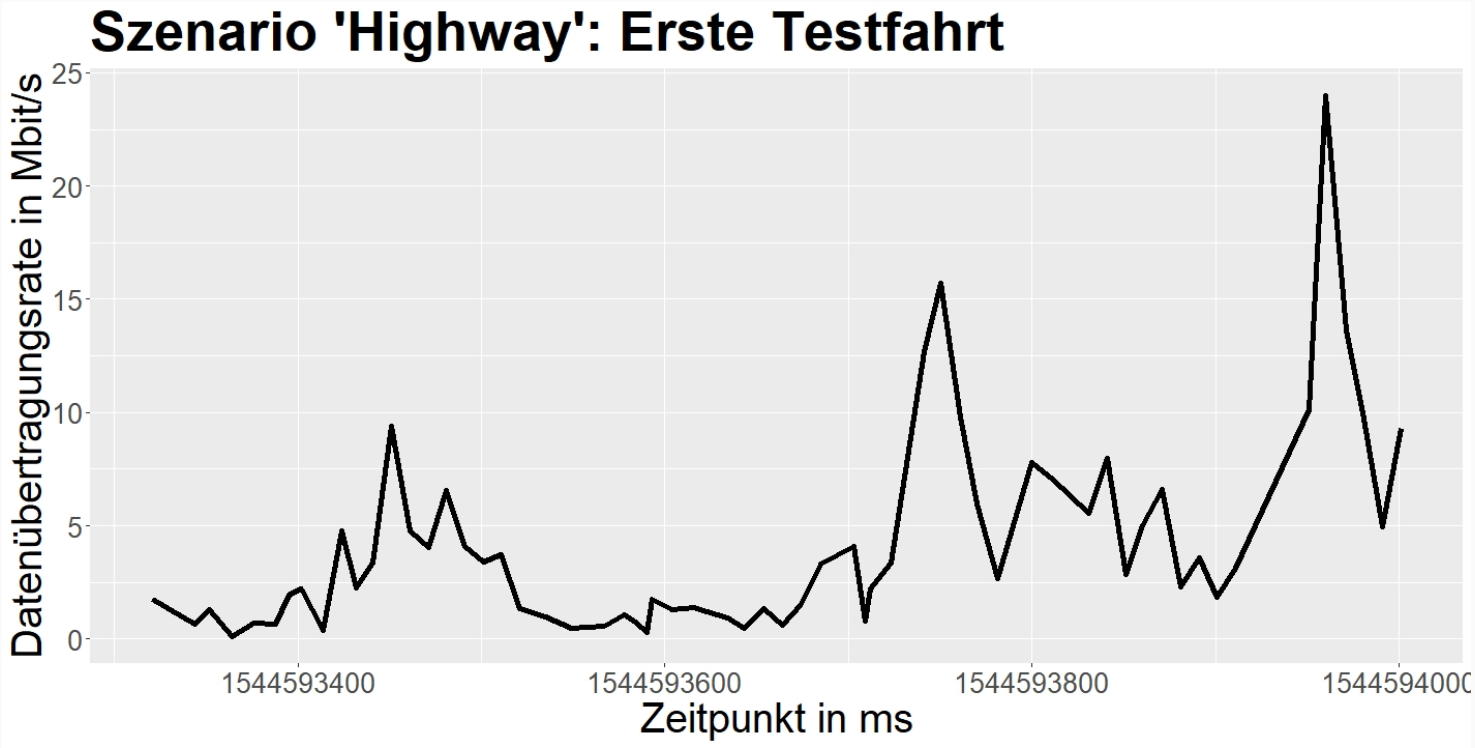


Figure 1: Grafik der auf der ersten Testfahrt im Szenario "Highway" gemessenen Datenübertragungsrate.

- Zeitreihe $y_1, ..., y_n$ (Zielvariable)
- k Zeitreihen $x_{i,1}, ..., x_{i,n}$ für $i = 1, ..., k$ (Einflussvariablen)

Regression mit ARMA-Fehlern

Lineares Regressionsmodell

$y_t = c + \beta_1 x_{1,t} + ... + \beta_k x_{k,t} + \epsilon_t$ mit Fehler ϵ_t und Konstante$c$

Annahmen an Fehler:

- $\forall t \in \{1, ..., n\} : E(\epsilon_t) = 0$
- $\forall s, t \in \{1, ..., n\} s \neq t : Cov(\epsilon_s, \epsilon_t) = 0$
- $Cov((\epsilon_1, ..., \epsilon_n)^T) = \sigma^2 \mathbb{1}_n$

Annahmen sind in unserer Situation nicht einhaltbar!

Regression mit ARMA-Fehlern

ARMA(p, q) Zusammengesetzes Modell aus

- $AR(p)$ (Auto Regressive):
  $y_t = c + \phi_1 y_{t-1} + ... + \phi_p y_{t-p} + e_t$ mit Fehler e_t und Konstante$c$
- $MA(q)$ (Moving Average): $y_t = c + e_t + \theta_1 e_{t-1} + ... +$
  $\theta_q e_{t-q}$ mit White Noise $e_t, e_{t-1}, ..., e_{t-q}$ und Konstante c

Zusammengesetzt:

$$y_t = c + \underbrace{\phi_1 y_{t-1} + ... + \phi_p y_{t-p}}_{AR(p)} + \underbrace{\theta_1 e_{t-1} + ... + \theta_q e_{t-q}}_{MA(q)} + e_t$$

Regression mit ARMA-Fehlern

Anwendung auf Regressionsfehler

Erinnerung: Fehler ϵ_t des linearen Modells sind autokorreliert $\Rightarrow$ erfüllen Voraussetzungen nicht

Lösung: Wende ARMA-Modell auf Fehler an

$$\epsilon_t = c + \phi_1 \epsilon_{t-1} + ... + \phi_p \epsilon_{t-p} + \theta_1 e_{t-1} + ... + \theta_q e_{t-q} + e_t$$

Modellgleichung Regression mit ARMA-Fehlern:

$$y_t = c + \sum_{i=1}^{k} \beta_i x_{i,t} + \underbrace{\sum_{j=1}^{p} \phi_j \epsilon_{t-j}}_{\text{vergangene Fehler LM}} + \underbrace{\sum_{k=1}^{q} \theta_k e_{t-k}}_{\text{vergangene Fehler ARMA}} + e_t$$

Regression mit ARMA-Fehlern

h-Schritt Punktvorhersage

- Ersetze Beobachtungen zu zukünftigen Zeitpunkten mit deren Vorhersagen
- Ersetze Fehler an vergangenen Zeitpunkten durch das entsprechende Residuum
- Ersetze Fehler an zukünftigen Zeitpunkten durch 0

Beispiel: $h = 2, k = 1, p = 2, q = 2$

$$y_t = c + \beta_1 x_t + \epsilon_t \text{ mit } \quad \epsilon_t = \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \theta_1 e_{t-1} + \theta_2 e_{t-2} + e_t$$

$$\widehat{y_{t+1}} = c + \beta_1 x_t + \widehat{\epsilon_{t+1}} \text{ mit } \widehat{\epsilon_{t+1}} = \phi_1 \epsilon_t + \phi_2 \epsilon_{t-1} + \theta_1 e_t + \theta_2 e_{t-1} + \underbrace{\widehat{e_{t+1}}}_{=0}$$

$$\widehat{y_{t+2}} = c + \beta_1 x_t + \widehat{\epsilon_{t+2}} \text{ mit } \widehat{\epsilon_{t+2}} = \phi_1 \widehat{\epsilon_{t+1}} + \phi_2 \epsilon_t + \theta \underbrace{\widehat{e_{t+1}}}_{=0} + \theta e_t + \underbrace{\widehat{e_{t+2}}}_{=0}$$

Validierung

k-fache Kreuzvalidierung

- beachtet Abhängigkeit der Datenpunkte nicht
- zerstört zeitliche Komponente
- verwendet eventuell zukünftige Beobachtungen für Prognose der Gegenwart

$\Rightarrow$ **Kreuzvalidierung für Zeitreihen**

Validierung

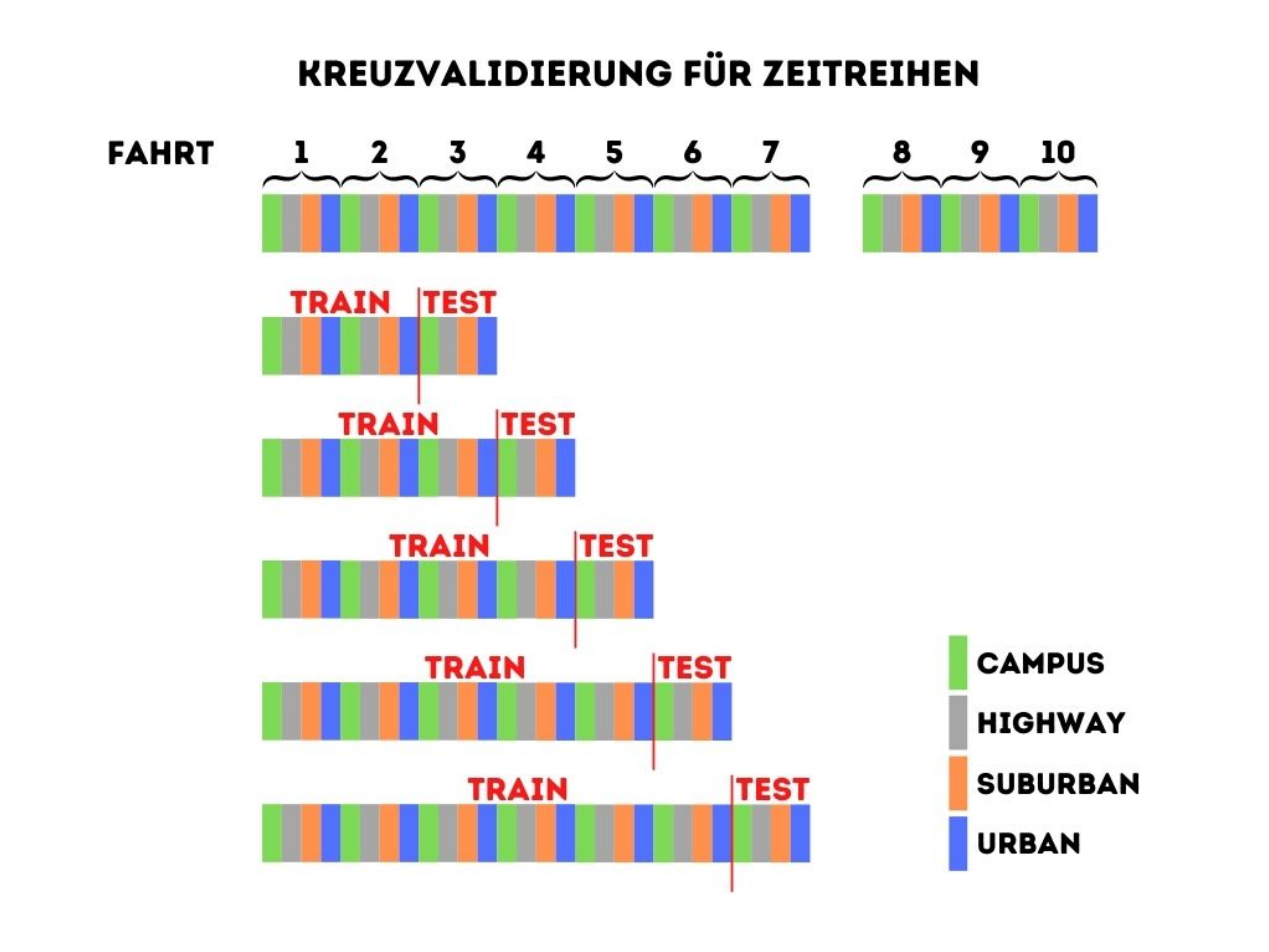


Figure 2: Grafik der auf der ersten Testfahrt im Szenario "Highway" gemessenen Datenübertragungsrate.

Task I: Data Rate Prediction

Validierung

frame

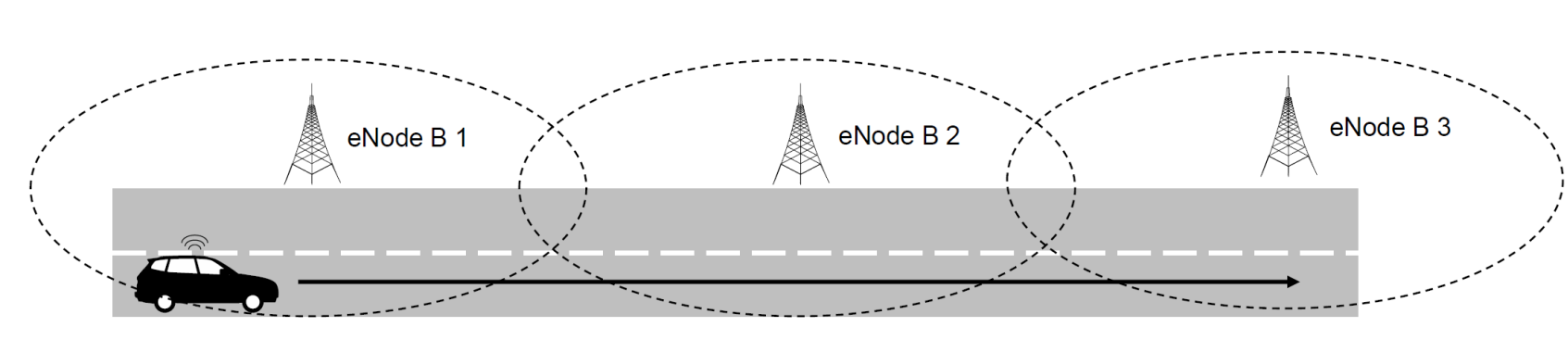
hallo

Task II - Handover Vorhersage und Link Lifetime

Aufgabenstellung Task II

Vorhersage des Handovers und Link Lifetime

- Vergleich des RSRP Wertes zur verbundenen Zelle sowie zu den Nachbarzellen
- Vorhersage des Handovers durch Angabe der Link Lifetime



Task II - Handover Vorhersage und Link Lifetime

Lösungsansatz Task II

Lösungsansatz Task II

Idee: Prädiktionsmodell für Link Lifetime mit Einfluss des RSRP der verbunden sowie der Nachbarzellen
$\rightarrow$ Datentransformation nötig

- Anpassen der RSRP Messwerte in "Cells" an RSRP Werte in "Context"
- Cell Id $\rightarrow$ eNodeB
- eNodeB Wechsel $\rightarrow$ Response Variable Link Lifetime

time_s	rsrp_dbm	ci	scenario	provider	enodeb	drive_id	rsrp_neighbor	link_lifetime
0.06	-98	13828122	campus	o2	54016	1	-99	18.01
1.07	-101	13828122	campus	o2	54016	1	-104	17.00
2.07	-101	13828122	campus	o2	54016	1	-104	16.00
3.07	-94	13828122	campus	o2	54016	1	-100	15.00
4.07	-94	13828122	campus	o2	54016	1	-100	14.00

Prädiktionsmodell Task II

- Anwendung des Prädiktionsmodells XGBoost um Link Lifetime vorherzusagen
- Validierung analog zu Task I mit Zeitreihenkreuzvalidierung

Irgendwas zum Schluss

Literatur i


T. Chen and C. Guestrin.
Xgboost: A scalable tree boosting system.
CoRR, abs/1603.02754, 2016.


T. Hastie, R. Tibshirani, and J. Friedman.
The elements of statistical learning: data mining, inference and prediction.
Springer, 2 edition, 2009.


B. Sliwa and C. Wietfeld.
Data-driven network simulation for performance analysis of anticipatory vehicular communication systems.
IEEE Access, 7:172638–172653, 2019.