Flight Level Prediction with a Deep Feedforward Network

SESAR Innovation Days 2018

Dr. Matthias Poppe, Debora Fieberg, Roland Scharff, Jörg Buxbaum

04.12.2018

member of FABEC

DFS Deutsche Flugsicherung

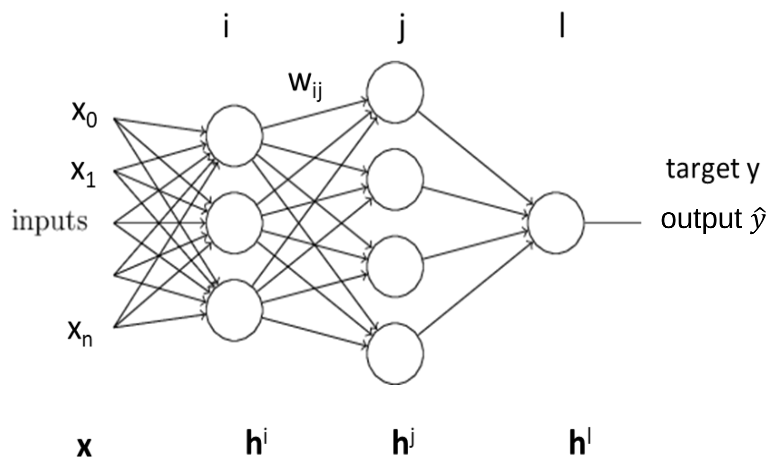
Background

- Controller decision support tools (DST) like Conflict detection rely on accurate 4D Trajectory Prediction (TP)
- Lateral (2D) position can be predicted quite accurately
  - After acceleration phase, aircraft operate with constant Calibrated Airspeed or Mach number; RNP enables accurate 2D navigation
- High uncertainty in vertical speed -> climb rate
  - Depends on many factors (not exhaustive): atmospheric conditions, aircraft weight, cost index, Flight Management System, operational restrictions, human intervention as controller clearance, pilot discretion
- To cater for uncertainty, DST need to apply buffer
  - Example: Controller Assistance Tools (CATO) use a static vertical buffer of ±500 feet/minute related to observed actual rate

Motivation

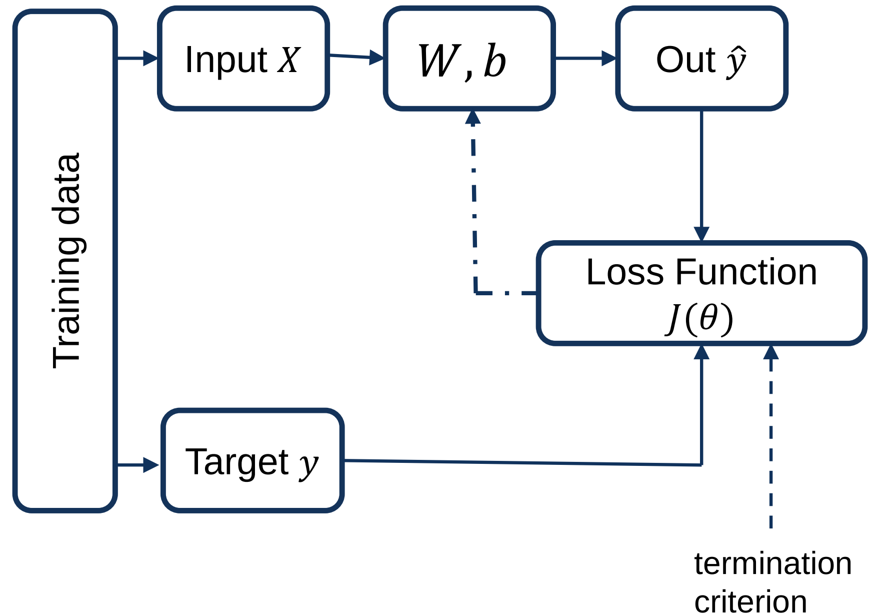
- For a prediction horizon of 6 minutes (typical value for short sectors with mostly vertical traffic patterns):
  - Buffer is ±3000 feet (60 Flight Level FL)
- Idea: develop and train a Deep Feedforward network, using only easily available data from Mode S plus aircraft type
  - Main objective: in the climb phase, for each departure, predict the reached flight level after n minutes (here: n=6)
  - Use this information to reduce buffer size for vertical TP and thus reduce number of false positives for conflict detection
- Once trained, each new departure flight can be used to further improve the neural network (online learning)

Feedforward Network (principle)



$$\boldsymbol{h^j} = \sigma(\boldsymbol{b^j} + W^j \boldsymbol{h^i}) \qquad (1)$$

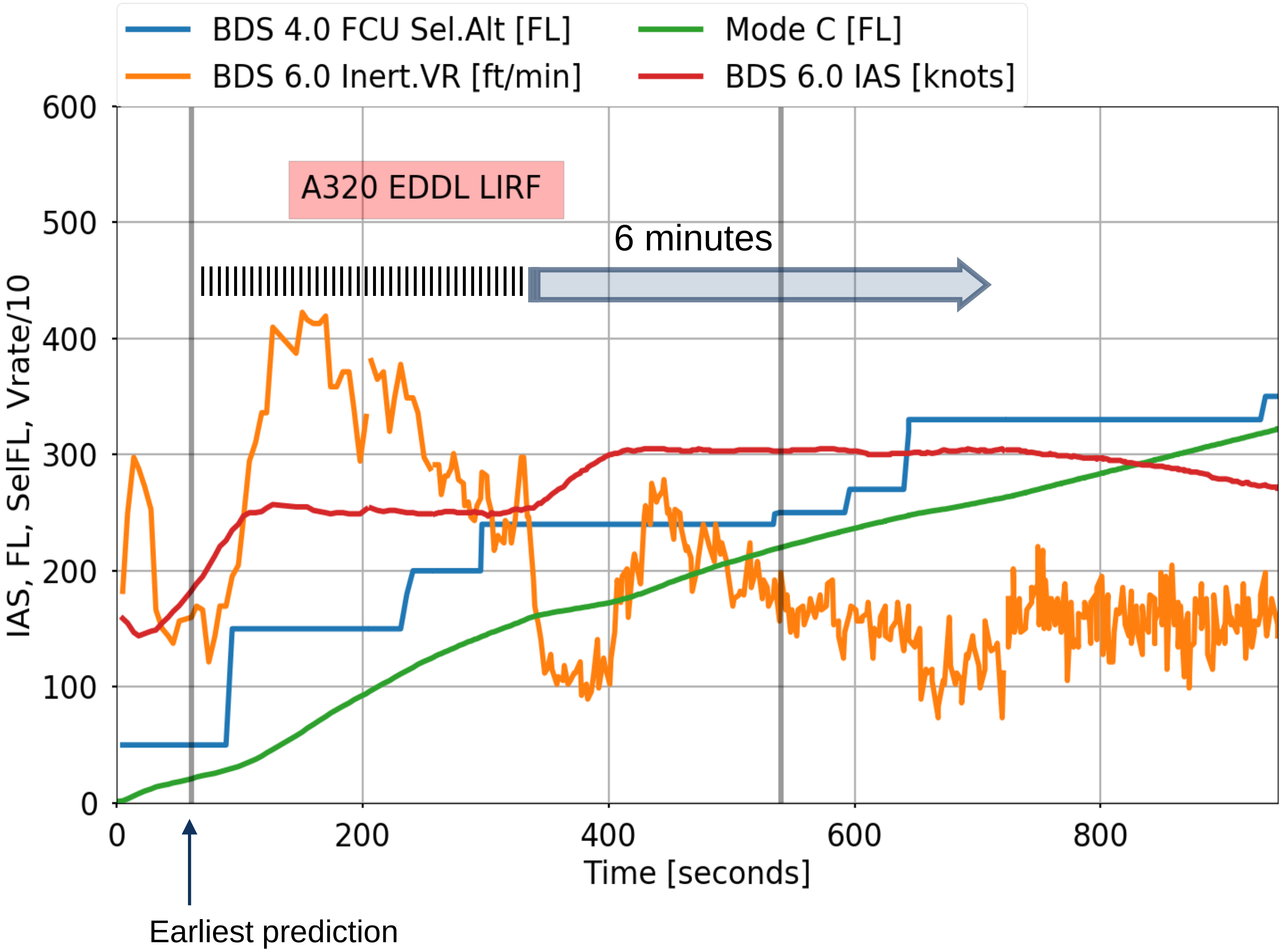
σ activation function



- Use supervised learning for regression task, i.e. predict FL to be reached in n minutes (target y) as numerical value
- Training data: update weight W, bias b of the neural network such that the loss function $J(\theta)$ will be minimized (e.g. mean squared error)
- Validation data will be used to evaluate the trained model

Raw Input Data

- Mode S Enhanced Surveillance (EHS) BDS Register (mandatory for IFR flights): 4.0 Selected Vertical Intention, 5.0 Track and Turn Report, 6.0 Heading and Speed Report (from ASTERIX CAT048)



Data Preparation

- Data cleansing -> select climbing flights
  - $min_t\,\{FL(t)\} < 20,\ max_t\,\{FL(t)\} > 285$ and $argmin_t\,\{FL(t)\} <\ argmax_t\,\{FL(t)\}$
- Extracted (raw) data for selected flights

Index	Abbreviation	Explanation	Unit
0	t	Time	sec
1	FL	Flight Level	FL
2	AID	Callsign	-
3	TAS	True Airspeed	kt
4	FCU	Flight Control Unit Selected Altitude	ft
5	GS	Ground Speed	kt
6	α	True Track Angle	deg
7	h	Magnetic Heading	deg
8	IAS	Indicated Airspeed	kt
9	Ma	MACH Number	Mach
10	r_{inert}	Inertial Vertical Speed (caveat: sign!)	ft/min
11	p	Barometric Pressure	mb
12	r_{baro}	Barometric Altitude Rate (caveat: sign!)	ft/min
13	r_{calc}	Calculated Rate of Climb	ft/min

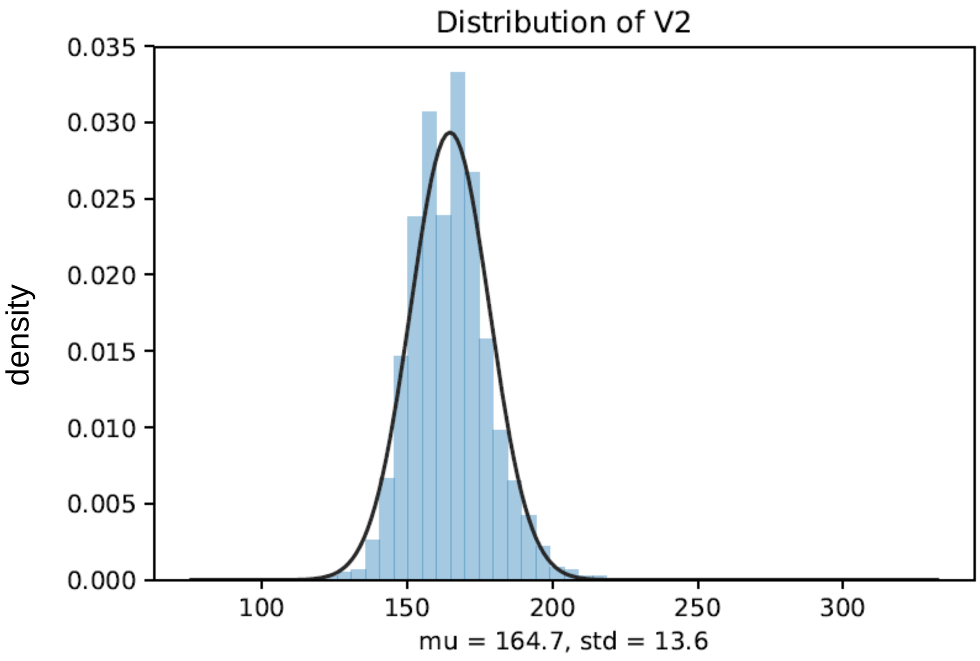
Derived Features

- From the raw input data, we derived several features x_0 to x_{19} which are supposed to impact the climb behavior as input for the neural network

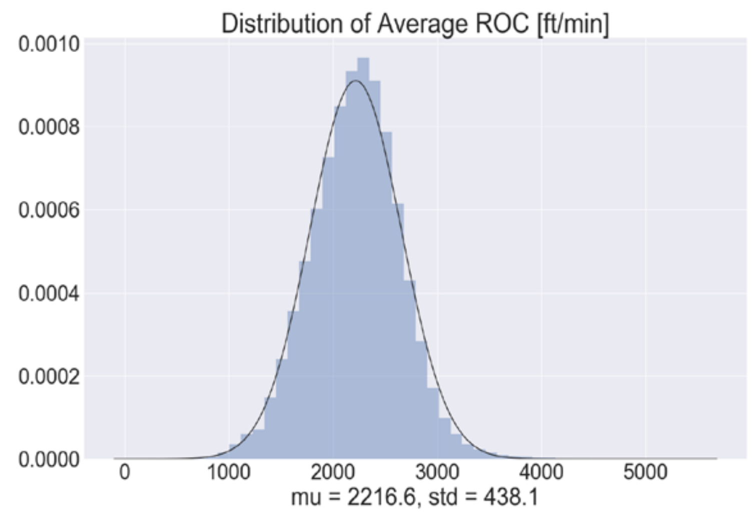
Feature	Explanation
x_0	Take Off Safety Speed V_2 (slide 8)
x_1, x_2	Average and actual Rate of Climb (slide 9)
x_3	Initial peak Rate of Climb (slide 10)
x_4, x_5	Wind component in flight direction
x_6	Heading Change flag (above/below threshold)
x_7	Altitude Restriction Flag (slide 11)
x_8	Time since FL100
x_9 to x_{13}	Flight Level 60, 45, 30, 15, 0 seconds ago
x_{14} to x_{18}	IAS 60, 45, 30, 15, 0 seconds ago
x_{19}	Aircraft type (slide 12)

Take Off Safety Speed

- Minimal speed aircraft climbs to first safe altitude
  - With all engines operative ~ V_2 + 10 knots
  - Depends on aircraft weight, atmospheric conditions
  - $IAS(t_x)\ with\ t_x\ such\ that\ FL(t_x)\ \leq 10$
  - Varies significantly due to aircraft parameters and among aircraft types

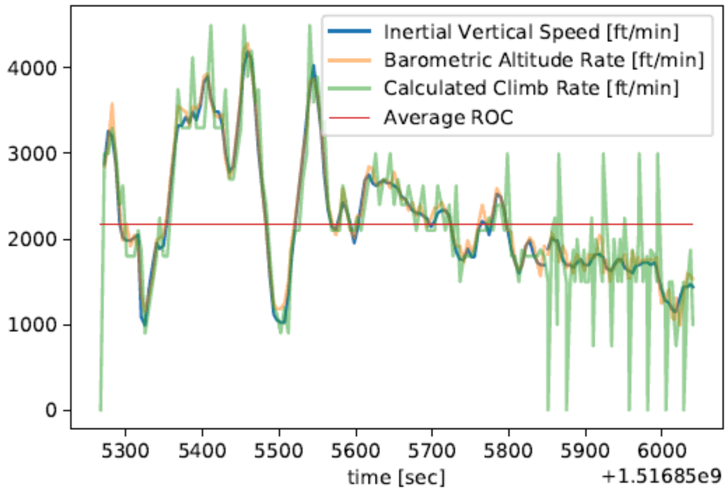


Average Rate of Climb



Available rates of climb:

- inertial vertical velocity ROC_{ins}
- barometric altitude rate ROC_{baro}
- calculation from flight level and time ROC_{calc}



$$x_2 := \frac{\sum_{t=0}^{t_x} \mathrm{ROC}(t)}{t_x} \text{ with } FL(t_x) < 285$$

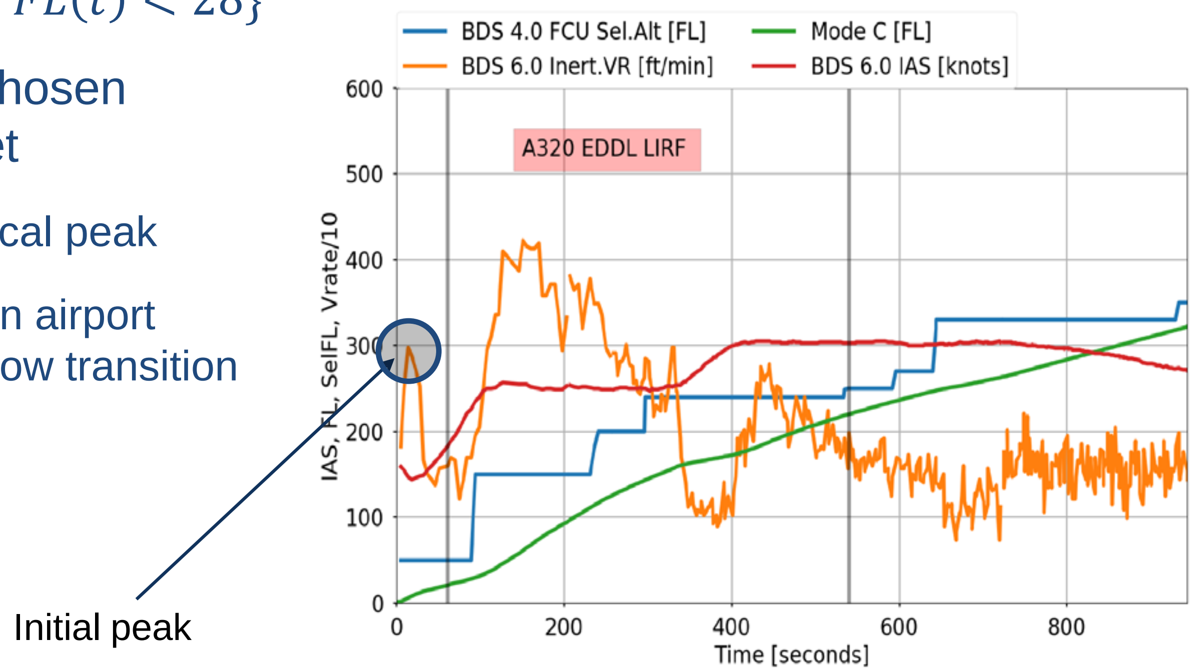
$$\text{with } ROC(t) = \begin{cases} ROC_{\mathrm{baro}}(t) & \text{, if } \mathrm{prop}_{\mathrm{ins}} \geq 0.8 \\ ROC_x(t) & \text{, else} \end{cases},$$

$$\text{for } x = \operatorname{argmax}\{\mathrm{prop}_x \mid x \in \{\mathrm{baro}, \mathrm{ins}\}\}$$

and prop_x proportion of finite datapoints.

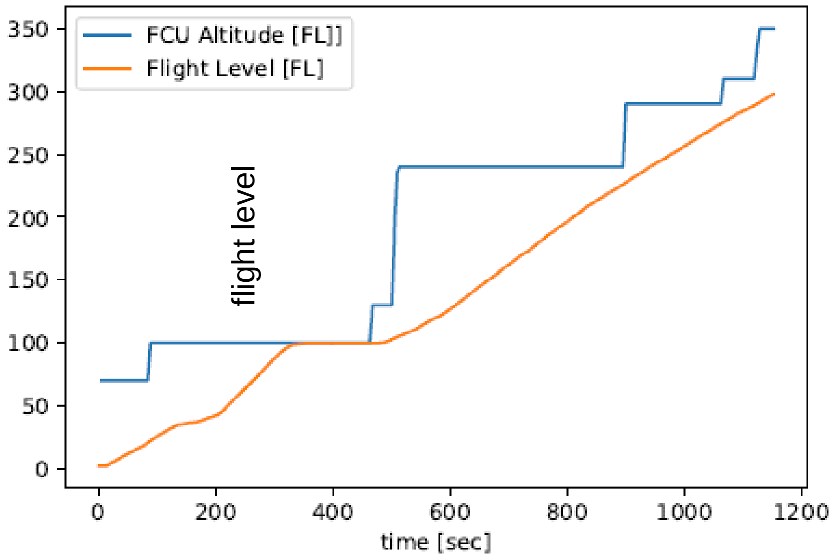
Initial peak Rate of Climb

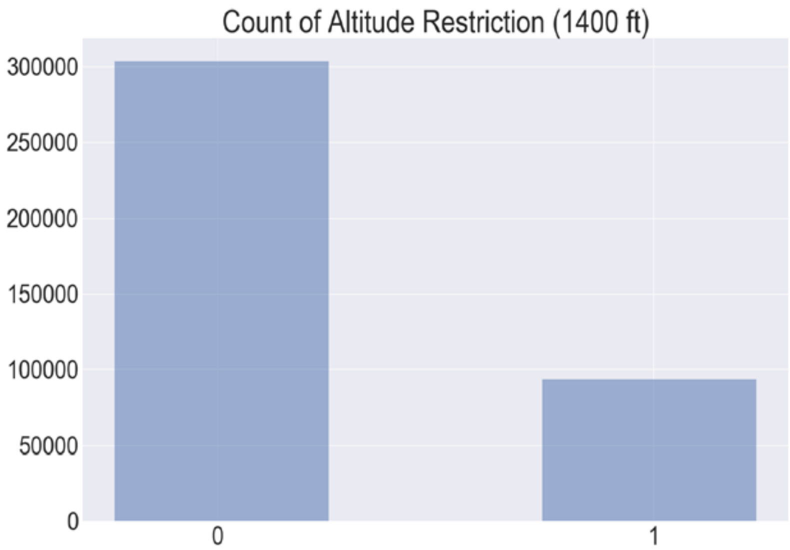
- Initial peak typical for performance of aircraft (?)
- (weak) correlation of peak and average ROC from FL100 to FL285: r = 0.40
- might be a hint to the energy share factor
- $x_3 := \max\{ROC(t) \mid FL(t) < 28\}$
- Value of 2800 feet chosen according to data set
  - Should include typical peak
  - Caveat: depends on airport height because below transition level



DFS Deutsche Flugsicherung

Altitude Restriction Flag

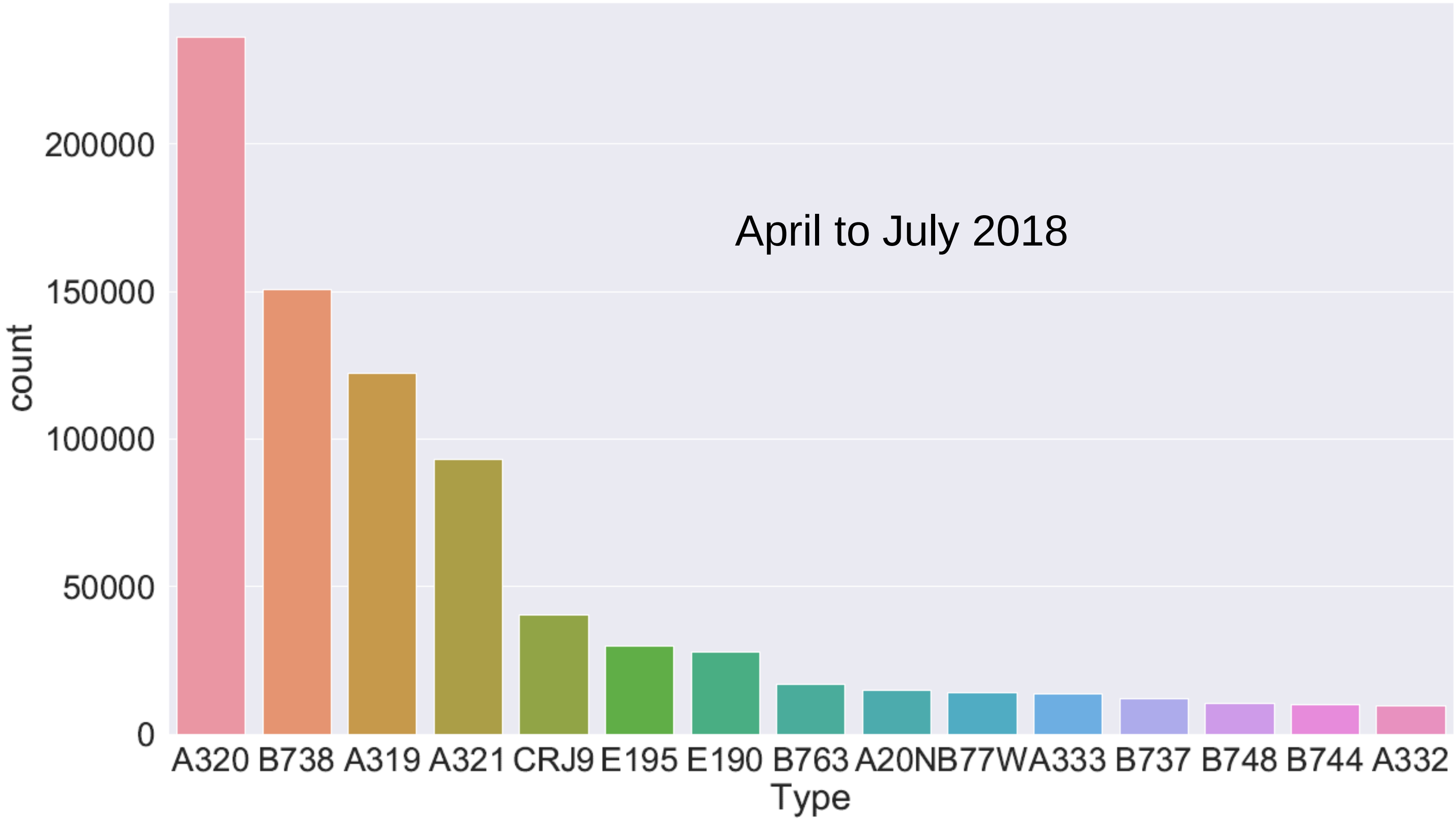




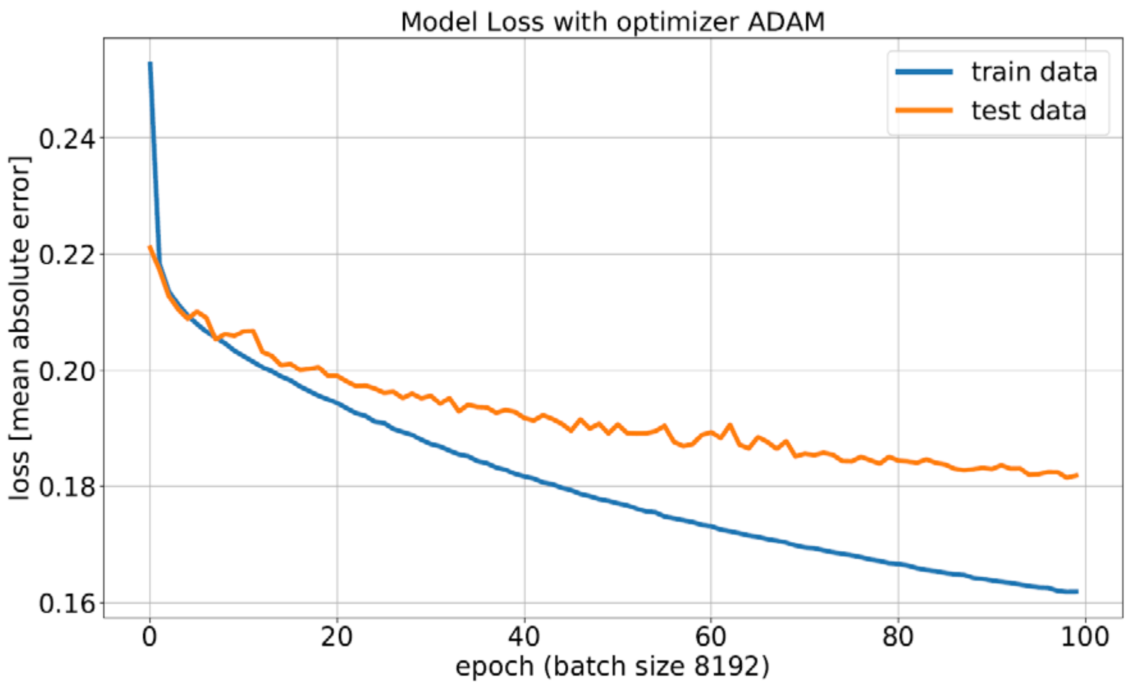
- Boolean flag $x_7 := \begin{cases} 0, & if\ \exists\ t\!: FCU(t) - 100 * FL(t) \leq \ \omega \\ 1, & else \end{cases}$
- We choose to set ω := 1400 [feet]
- Majority of flights climb „unrestricted“ according to this criterion

DFS Deutsche Flugsicherung

Aircraft type (all airports)



Training: Model Loss



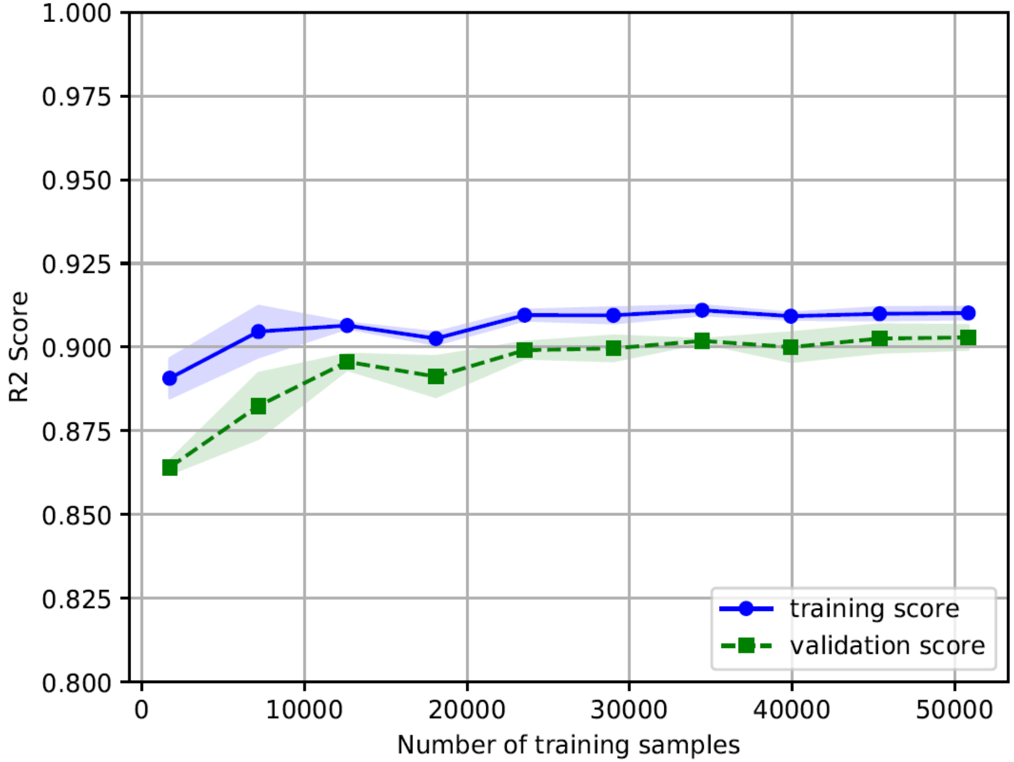
Adam = Adaptive moment estimation optimizer (extension of Stochastic Gradient Descent)

- Training after 100 epochs shows still decrease in loss for both train and test data

- Deep feedforward network
  - Keras with tensorflow and GPU
  - N=7 layers, M~4000 neurons
  - ReLu activation function for hidden layers
  - Different optimizer tested -> uncritical if not stucked at the beginning
  - Dropout regularization with 15% to 30% drop rate

DFS Deutsche Flugsicherung

Learning Curve



$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$ coefficient of determination

$$SS_{\text{res}} = \sum_i (y_i - f_i)^2$$ Predicted data ($\hat{y}$)

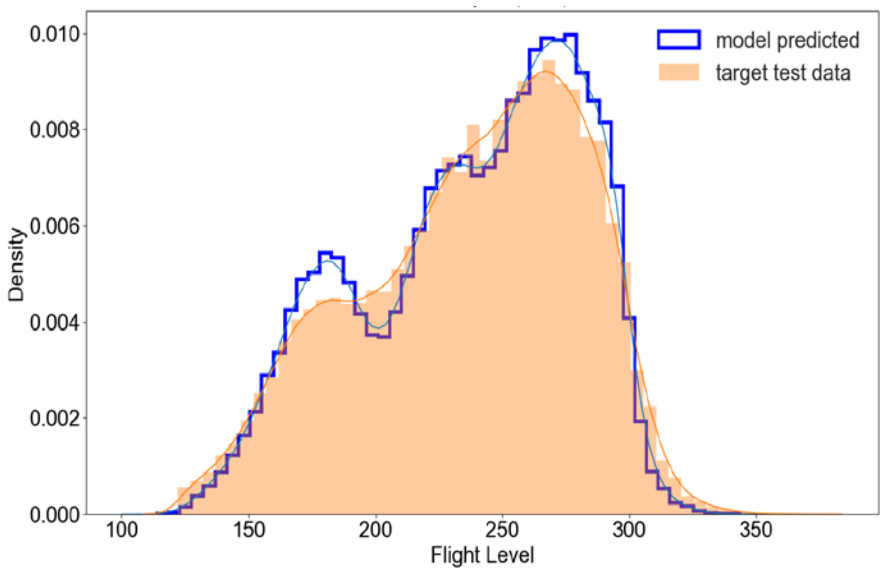
$$SS_{\text{tot}} = \sum_i (y_i - \bar{y})^2$$ Mean of observed data

- About 100.000 climbing flights available from all German airports
- For training, 4 samples used per flight
  - 400k features/labels
- Learning curve suggests that at least 50k samples are required
  - Even more if network becomes „deeper“, i.e. more layer or more neurons per layer
- R^2 is the proportion of the variance in the dependent variable that is predictable (max = 1)

Results

Metric	Prediction Time 360 seconds	
	Test Data	Train Data
R^2 score	0.922 (0.931)	0.931 (0.942)
Mean error	8.14 (7.58) FL	7.63 (6.78) FL
Std. Dev. error	7.21 (6.84) FL	7.04 (6.56) FL

Metric	Prediction Time 180 seconds	
	Test Data	Train Data
R^2 score	0.975 (0.978)	0.978 (0.981)
Mean error	5.35 (5.13) FL	4.95 (4.53) FL
Std. Dev. error	5.32 (5.16) FL	5.13 (4.85) FL



() in brackets: only aircraft type A320 selected
1 FL = 100 feet

- Feedforward network is able to learn distribution $p(y|\boldsymbol{x})$
- Mean absolute error about 800 feet for six minutes prediction horizon with standard deviation of abuot 700 feet
- If only specific aircraft type (here: A320) selected, results are getting better
- The shorter the time horizon, the better the result

Conclusions

- Results confirm possibility to reduce uncertainty buffer for TP in climb phase
- Network is able to predict flight level with supervised learning and can „distinguish“ different operational environments due to training data
- Most effort put in extracting features (specialist know-how)
  - Building network with Keras, tensorflow straightforward although optimization of hyperparameter needs lot of computing power
- Number of features can possibly be modified and/or reduced
  - Apply Sequential Backward Selection algorithm
  - Cluster aircraft types with similar (climb) performance
- Further improvements envisaged by
  - Bootstrap aggregating
  - Combination with Recurrent Architecture, e.g. Long Short Term Memory in order to take into account dynamics of climb phase

Future Work

- Analysis of outliers ongoing
  - First analysis suggests that biggest deviations are caused by level-off segments because of controller clearances
  - Train network only for unrestricted climbs and include clearance as a feature?
- Compare data to Full Flight Simulator
  - Controlled enviroment with known parameter yet realistic flight behavior
  - Derive features and predict flight level with trained feedforward network for this specific aircraft type
  - May allow to estimate parameters like Cost Index, Take Off weight
- Extend network architecture to allow online-prediction for departing flights
  - Each flight can in turn be used to further train and improve the network

Thank you very much for your attention!

DFS Deutsche Flugsicherung

Backup: Keras model summary()

```
Layer (type)            Output Shape         Param #
================================================
dense_1 (Dense)          (None, 2460)          415740
dense_2 (Dense)          (None, 1220)          3002420
dense_3 (Dense)          (None, 610)          744810
dense_4 (Dense)          (None, 100)          61100
dense_5 (Dense)          (None, 40)          4040
dense_6 (Dense)          (None, 10)          410
dense_7 (Dense)          (None, 1)          11
============================================================
Total params: 4,228,531
Trainable params: 4,228,531
Non-trainable params: 0
```

Backup: Post results

BATCH_SIZE = 8192

EPOCHS = 200+ReduceLRonPlateau

PREDICTION_TIME: 360 seconds

ALL_ AIRCRAFT _TYPES

Test set score (loss) = 0.1817

r2 score test : 0.918

r2 score train: 0.922

Test Data mean absolute error: 7.28 FL

Test Data error standard dev =   6.90 FL

Train Data mean absolute error: 6.91 FL

Train Data error standard dev =   6.80 FL