Machine Learning (2022) 111:243–272
https://doi.org/10.1007/s10994-021-05957-0

Boosting Poisson regression models with telematics car driving data

Guangyuan Gao[1] · **He Wang**[2] · **Mario V. Wüthrich**[3]

Received: 9 May 2020 / Revised: 1 February 2021 / Accepted: 9 February 2021 /
Published online: 21 March 2021

Abstract
With the emergence of telematics car driving data, insurance companies have started to boost classical actuarial regression models for claim frequency prediction with telematics car driving information. In this paper, we propose two data-driven neural network approaches that process telematics car driving data to complement classical actuarial pricing with a driving behavior risk factor from telematics data. Our neural networks simultaneously accommodate feature engineering and regression modeling which allows us to integrate telematics car driving data in a one-step approach into the claim frequency regression models. We conclude from our numerical analysis that both classical actuarial risk factors and telematics car driving data are necessary to receive the best predictive models. This emphasizes that these two sources of information interact and complement each other.

Keywords Densely connected feed-forward neural network · Convolutional neural network · Combined actuarial neural network · Claims frequency modeling · Telematics car driving data · Poisson regression · Generalized linear model · Regression tree · Telematics heatmap

1 Introduction

Product development and car insurance pricing is an important task of actuarial modeling. Generalized linear models (GLMs) are state-of-the-art for car insurance pricing. To overcome some of the shortcomings of GLMs other regression models are also used, e.g., generalized additive models (GAMs) and regression trees are promoted in Henckaerts et al.

Editors: Tim Verdonck, Bart Baesens, María Óskarsdóttir and Seppe vanden Broucke.

✉ Mario V. Wüthrich
mario.wuethrich@math.ethz.ch

[1] Center for Applied Statistics and School of Statistics, Renmin University of China, 100872 Beijing, China

[2] Department of Finance and Ying Shang Nan Ke Actuarial Science Center, Southern University of Science and Technology, 518055 Shenzhen, China

[3] Department of Mathematics, ETH Zurich, RiskLab, 8092 Zurich, Switzerland

(2018), boosting models are used in Henckaerts et al. (2019), Lee (2021) and Yang et al. (2018), or neural networks are proposed in Ferrario et al. (2018). Many pricing approaches have in common that they are solely based on classical policyholder information such as age of driver, type of car, age of car, vehicle power, etc. This classical policyholder information is called a priori information because it is available at the conclusion of contract, see Verschuren (2021). Increasingly, posterior information about individual policyholder behavior and insurance claims is collected, and this a posteriori information is incorporated in insurance policy renewals. In car insurance, a posteriori information is often encoded in a bonus-malus system (BMS) which scores past claims history and directly affects the insurance prices of policy renewals by a multiplicative factor. One stream of literature on BMS studies optimal design, efficiency and economic questions related to BMS, see Loimaranta (1972), De Pril (1978), Lemaire (1995), Denuit et al. (2007), Brouhns et al. (2003) and Ágoston and Gyetvai (2020). A second stream of literature rather addresses the question of how an existing BMS can be used to improve the predictive power for forecasting future claims since a BMS reveals past policyholder behavior, see e.g., Boucher and Inoussa (2014), Boucher and Pigeon (2018) and Verschuren (2021).

With the emergence of telematics car driving data one can even go one step further, namely, one does not only have a discrete claim indicator variable (which runs into the BMS), but insurers receive continuously personalized car driving information about their policyholders. This continuous telematics data may reveal that a specific young driver has a cautious driving style, while a matured driver may still have a wild and aggressive driving behavior. Telematics car driving data will encode such differences. Our goal is to explore first steps on how such information can be extracted from telematics car driving data. This is far from being trivial because telematics car driving data comes with all its challenges such as big data (we typically have TBs of data that needs to be processed), data error, etc.

Our telematics data records speed and acceleration in all directions second by second from the start of the engine to the switch off of the engine. Current literature proposes three different ways to perform feature engineering on such type of data: (a) Weidner et al. (2016, 2017) extract covariates from time series of telematics data using discrete Fourier transforms; (b) Huang and Meng (2019), Paefgen et al. (2014), Sun et al. (2020) and Verbelen et al. (2018) do not directly consider telematics car driving data in time series structure, but rather calculate scores such as average speed, 90%-quantile of acceleration rates, or proportions of driving distances on different types of roads; (c) Gao et al. (2019), Gao and Wüthrich (2019) extract covariates from speed-acceleration heatmaps using principal components analysis (PCA) and neural network architectures. These papers have in common that they first extract several potential risk factors from telematics data, and then select the useful ones in a second step using a regression model for claim frequency modeling. This procedure has limitations because it assumes that the extracted feature information is the relevant one, i.e., it involves judgment in a first step similar to manual feature engineering, which may not be optimal for subsequent steps.

We mention other literature on telematics data which extracts specific information. Ayuso et al. (2016a, b, 2019), Boucher et al. (2017) and Lemaire et al. (2016) study risk exposures such as driving distances or time exposures. Such information is interesting for two reasons. Firstly, an appropriate exposure acts as an offset in regression modeling and, henceforth, does not need explicit modeling. Secondly, new insurance products are developed where prices are calculated on a pay-as-you-drive (PAYD) basis. Such products may also be interesting from an environmental point of view because driving less makes insurance less costly. Denuit et al. (2019) propose a credibility model to incorporate posterior information of driving behavior, this is in the spirit of BMS. Richman (2020a, b) discusses

possible ways to analyze telematics data. Ho et al. (2014), Hung et al. (2007) and Kamble et al. (2009) study telematics data and driving cycles to understand vehicular emissions, energy consumption and impacts on traffic in different cities of the world. The techniques used for these studies rely on similar tools as proposed in Gao et al. (2019).

In this paper, we use the speed-acceleration heatmap construction proposed in Gao et al. (2019) as a representation of driving behavior. However, in contrast to this former paper, we do not use a two-step approach by first scoring heatmaps and then using these scores in a regression analysis. In this paper, we conduct both feature engineering of the speed-acceleration heatmap and claim frequency regression simultaneously, using a densely connected feed-forward neural network and a convolutional neural network, respectively. That is, the networks learn a feature representation of the speed-acceleration heatmap that is directly used in a Poisson regression model. We start from a densely connected feed-forward neural network because this is the most basic neural network. Secondly, we challenge the previous approach with a convolutional neural network. Our data has a spatial structure, and it is known that convolutional neural networks can deal very successfully with spatial objects; we refer to Goodfellow et al. (2016) for a general discussion of neural networks and to Chollet and Allaire (2018) for an introduction to the neural network package `Keras` used in this paper. The fundamental difference between the two types of neural networks is that dense layers learn global patterns on the input space, while convolution layers learn local patterns. Compared to dense layers, convolution layers have relatively fewer parameters since they apply the same convolutional operation to different local regions of the input space. Their main property is translation invariance which allows convolution layers to find similar patterns at different locations of the input space, see Wiatowski and Bölcskei (2018), Zhang et al. (1988) and Zhang et al. (1990). It is often said that deep learning models are black boxes, but this is not necessarily true for convolutional neural networks. Convolutional neural network can be interpreted and we discover patterns in the speed-acceleration heatmaps which explain the most relevant drivers of higher claim frequencies.

In this paper, we are going to compete three different set-ups: (1) Poisson GLM based on classical actuarial risk factors (covariates); (2) Poisson neural network regression model based on telematics data, and (3) a combination of (1) and (2) in the spirit of Wüthrich and Merz (2019). From our numerical analysis we conclude that (3) is the most powerful approach. Firstly, not surprisingly, Poisson GLMs based on classical risk factors can be enhanced by telematics information, and secondly, telematics information is not sufficient because classical actuarial risk factors may reveal under which circumstances the telematics data has been generated. Thus, both sources of information interact which makes them a more powerful predictive tool.

1.1 Organization

Section 2 introduces our GLM approach for claim frequency modeling using classical risk factors. Section 3 describes our telematics car driving data and it establishes two neural networks for claim frequency modeling using speed-acceleration heatmaps. Section 4 considers a combined actuarial neural network for claim frequency modeling using both the classical risk factors and the speed-acceleration heatmaps. Moreover, the convolutional neural network solution is interpreted to explain how a driving behavior risk factor is constructed from speed-acceleration heatmaps. Section 5 concludes with our main findings.

2 Claims frequency modeling using classical risk factors

Our data considers compulsory motor third-party liability (MTPL) insurance of $n = 973$ cars in China. Each insurance policy has the same maximal coverage of CNY 122, 000. These MTPL insurance policies have been active within the time period from 01/01/2014 to 31/05/2017, and we have full reporting information up to 29/06/2017. A preliminary analysis indicates that more than 99% of all claims are reported within one month after the accident date. For this reason (and because information is missing), we neglect late reported claims after 29/06/2017; we expect that such late reported claims only marginally influence our analysis. For insurance policies $i = 1, \ldots, n$, we denote the response variable of claim counts by $Y_i \in \mathbb{N}_0$, the exposure of the effective policy duration by $e_i > 0$ (also called years-at-risk), and the set of classical risk factors by $\boldsymbol{x}_i$. We assume that the claim counts can be described by the following regression model

$$Y_i \overset{\text{ind.}}{\sim} \text{Poisson}\big(e_i \lambda(\boldsymbol{x}_i)\big), \qquad \text{for } i = 1, \ldots, n, \tag{2.1}$$

where $\lambda : \mathcal{X} \to \mathbb{R}_+$ is a regression function mapping the covariates $\boldsymbol{x}_i \in \mathcal{X}$ to expected frequencies $\lambda(\boldsymbol{x}_i) \in \mathbb{R}_+$. The general aim is to choose regression functions $\lambda(\cdot)$ such that they describe the systematic effects in the claim counts as accurately as possible. This choice involves pre-processing of covariates $\boldsymbol{x}_i \in \mathcal{X}$ (and choices of appropriate covariate spaces $\mathcal{X}$), which is part of regression modeling.

Before giving a descriptive analysis of our data, we give the main summary statistics. The total exposure over the entire portfolio is $\sum_{i=1}^n e_i = 2,177$ years-at-risk, i.e., several policies run over multiple years, the average exposure being $\sum_{i=1}^n e_i/n = 2.24$ years-at-risk. Figure 1 (top-left) shows a histogram of the aggregate exposures with policies partitioned w.r.t. the exposure lengths e_i. The overall (homogeneous) claim frequency estimate is $\bar{\lambda} = \sum_{i=1}^n Y_i / \sum_{i=1}^n e_i = 0.24$; this average is consistent with the market benchmark in China but it is much higher than typically in Europe and North America. In China, MTPL policies cover both physical injuries and property damages of third party regardless whether the policyholders is at fault or not, only the corresponding maximal coverage differs. This explains why the frequency is comparably higher than in other regions of the world. Figure 1 (top-right) shows the claim counts Y_i on each policy. Most policies do not suffer any claims, and very few policies have more than 3 claims.

2.1 Feature engineering

Our first modeling attempt for regression problem (2.1) is to choose a Poisson generalized linear model (GLM) with log-link function (being the canonical link for the Poisson GLM). This choice implies that covariates $\boldsymbol{x}_i$ impact the regression function linearly on the canonical scale, which, in turn, requires covariate pre-processing so that we receive reasonable regression models. We start by describing the available covariate information. We consider five covariate components: Chinese region ($\texttt{region}_i$), driver's gender ($\texttt{gender}_i$), driver's age ($\texttt{driver\_age}_i$), car's age ($\texttt{car\_age}_i$) and average driving hours in (0, 80] km/h per week ($\texttt{ave\_hours}_i$), for all insurance policies $i = 1, \ldots, n$. Our preliminary data cleaning ensures that the main driver of a car does not change over the entire observation period and we concatenate policy renewals of the same driver over this observation period. Thus, we can follow the same driver for at most 3 years and 5 months from 01/01/2014 to 31/05/2017; the resulting exposures e_i are shown in Fig. 1 (top-left). We remark that we

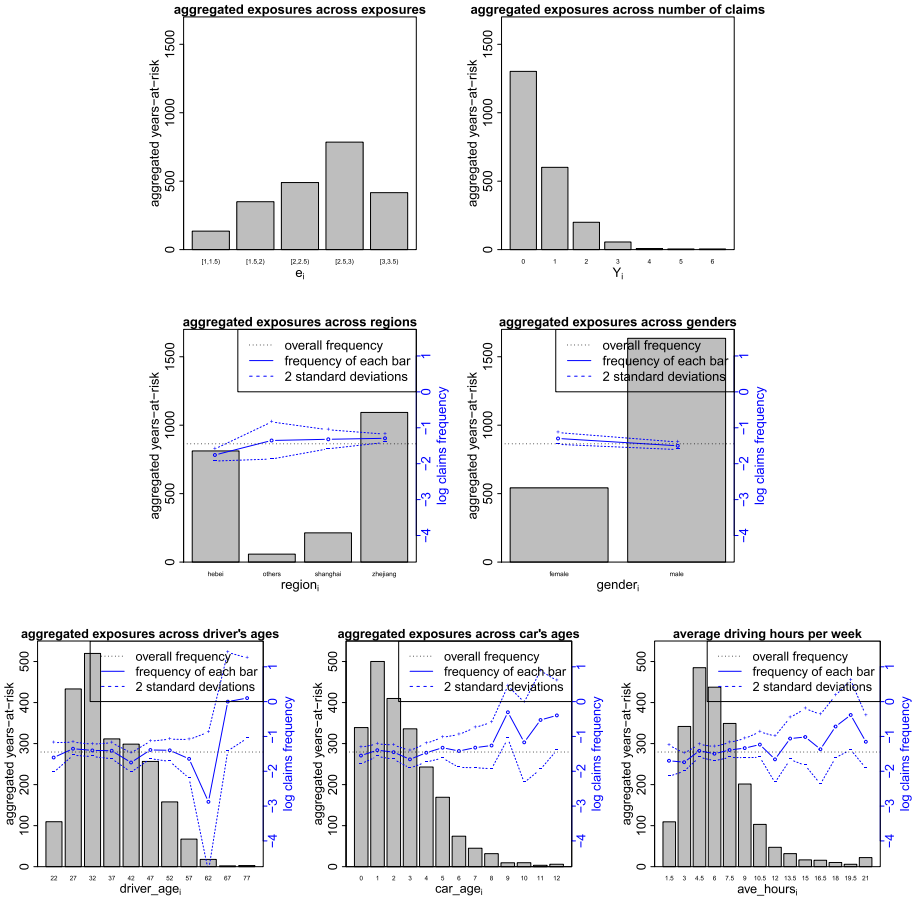

Fig. 1 Histogram of the distribution of exposures (left axis) and the corresponding logarithm of the empirical claim frequencies (right axis and in blue color, where appropriate): policies partitioned w.r.t. exposures e_i (top-left), claim counts Y_i (top-right), regions (middle-left), gender (middle-right), driver's age (bottom-left), car's age (bottom-middle), and average driving time (bottom-right); each row has identical y-scale

exclude several other covariates such as number of seats, car brand or price of car since a preliminary analysis indicates that those covariates are less significant for claims frequency prediction, at least for our small insurance portfolio, otherwise we may run into issues of over-fitting.

We are going to provide empirical statistics for these five covariates in Fig. 1 (middle and bottom rows). The distribution of the exposures is shown on the left axis (in black and gray bars), and the empirical frequencies are shown on the right axis (in blue color) with dotted blue lines giving estimated 2 standard deviation confidence bounds. We take the logarithm of the empirical claim frequencies because for small exposures they are volatile; moreover, the y-scales are identical on each row.

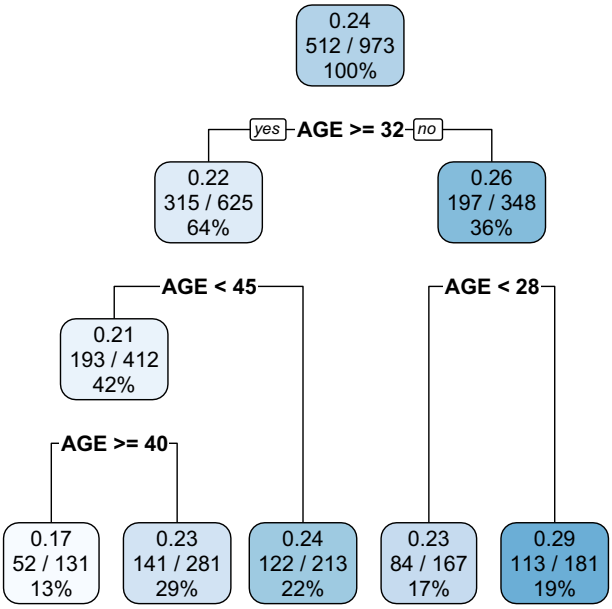

Fig. 2 Marginal Poisson regression tree for claim frequencies w.r.t. driver's age

2.1.1 Categorical variables: Chinese regions and driver's gender

In a preliminary step, we have merged 11 Chinese regions containing only very small exposures, resulting in four different regions $\{\texttt{Hebei}, \texttt{Others}, \texttt{Shanghai}, \texttt{Zhejiang}\}$. From Fig. 1 (middle-left) we observe that `Hebei` province has a substantially lower empirical frequency than the other 3 Chinese regions. Concerning gender, male drivers have a slightly lower empirical frequency, see Fig. 1 (middle-right), however, the difference not being significant on a 95% confidence level.

2.1.2 Driver's age

Typically, a suitable regression function for car claim frequency modeling is non-monotone in the driver's age variable. Therefore, the driver's age covariate needs pre-processing. There are two different ways to do so, either we use categorical coding by building age groups or we use a different functional form for driver's age, for instance, a natural cubic spline leading to a GAM. For the moment, we consider categorical coding of the driver's age variable. We explore a marginal Poisson regression tree using covariate $\texttt{driver\_age}_i$ as explanatory variable and exposure e_i as offset for building age groups of homogeneous claims frequency. This Poisson regression tree suggests to build 5 age groups to receive sufficient homogeneity within each age group (still keeping sufficient exposures in all age groups), see Fig. 2. We observe that the smallest group contains 13% of all policies and the biggest group 29% of all policies. The resulting age groups are given in Table 1.

We note that we will further merge age groups in Sect. 2.2. We have also explored incorporating the driver's age variable as a continuous covariate in a GAM. The predictive performance of the latter has been similar to the GLM with categorical coding but using a

Table 1 Chosen age groups

driver_age	20-27	28-31	32-39	40-44	45+
age_group	young_1	young_2	middle_1	middle_2	matured
number of policies	17%	19%	29%	13%	22%

more complex regression function, therefore, we work with the simpler age grouping version given in Table 1.

2.1.3 Car's age

For cars aged less than 3 years, the claim frequencies are similar. For cars aged more than 3 years, the logarithm of claim frequency has a linearly increasing shape, see Fig. 1 (bottom-middle). Therefore, we pre-process the car's age to create a new explanatory variables, $\texttt{car\_age} \mapsto \texttt{car} = \max(0, \texttt{car\_age} - 3)$.

2.1.4 Average driving time per week

As emphasized in Ayuso et al. (2016a, b, 2019), the total driving time is important covariate information. At this stage, it is debatable whether total driving time is a classical or a telematics covariate because this information is only available if suitable devices are installed in the cars. We remark that we follow insurance policies over multiple years, but only for the most recent periods there is telematics data available. For this reason, we typically have a longer observation period of claims history on insurance policies than of corresponding telematics data. As a compromise we calculate average driving time per week from the time periods where telematics data is available. An implicit assumption is that the calculated average driving time per week using the more recent periods of telematics data is a good approximation for the entire observation period of insurance exposure. Thus, we create a variable 'average driving time per week' (in hours), we cap this variable at 21 hours per week (to avoid outliers) and we only account for the time in speeds (0, 80]km/h, since this corresponds to the telematics heatmaps discussed below. From Fig. 1 (bottom-right), we observe a linear relationship between the logarithm of claim frequency and the average driving hours per week. Note that one can also use the average driving distance per week instead, both variables are measures of driving intensity.

2.2 A generalized linear model for claims frequency

The five classical risk factors have been pre-processed as described above to provide covariate

$$\boldsymbol{x}_i = \left(\texttt{region}_i, \texttt{gender}_i, \texttt{age\_group}_i, \texttt{car}_i, \texttt{ave\_hours}_i\right) \in \mathcal{X}.$$

The variable `gender` is binary. The age of car variable `car` and the average driving time per week `ave_hours` are continuous. The variable `region` and the age group variable `age_group` are incorporated by using dummy coding. Henceforth, we have 10-dimensional covariate space $\mathcal{X} \subset \mathbb{R}^{10}$ that can be directly used in a Poisson GLM with canonical log-link.

Table 2 Stratified partition w.r.t. empirical claim frequencies

Data	Number of drivers	Exposure	Number of claims	Frequency
Training data $\mathcal{D}_1$	584	1, 318	297	0.23
Validation data $\mathcal{D}_2$	196	438	109	0.25
Test data $\mathcal{D}_3$	193	421	106	0.25
Total	973	2, 177	512	0.24

We split our entire portfolio into a learning data set and a test data set in a stratified way w.r.t. empirical claim frequencies. The learning data set contains 780 cars with a total exposure of 1, 756 years-at-risk, and the test data set contains 193 cars with a total exposure of 421 years-at-risk. We calibrate our models on the learning data set and evaluate its out-of-sample predictive performance on the test set. In later sections on neural network regression models, we will further split the learning data set into training data set $\mathcal{D}_1$ and validation data set $\mathcal{D}_2$. The validation data set is used to determine hyper-parameters while training the models with the training data set. Denote the index sets of training, validation and test data sets by $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$, respectively. We summarize the partition used throughout this document in Table 2.

2.2.1 Poisson generalized linear model

We start with the Poisson GLM. The basic assumption is that the underlying expected claims frequency of λ has a multiplicative structure in the covariate components $\boldsymbol{x}_i$ providing regression function on the canonical scale (under log-link)

$$\boldsymbol{x}_i \;\mapsto\; \log \lambda(\boldsymbol{x}_i) = \beta_0 + \alpha_{\texttt{region}_i} + \gamma_{\texttt{gender}_i} + \delta_{\texttt{age\_group}_i} + \beta_1 \texttt{car}_i + \beta_2 \texttt{ave\_hours}_i. \tag{2.2}$$

We choose `male` driver, `middle_1` age group and `Zhejiang` region as reference levels for dummy coding. The coefficients are estimated by minimizing the Poisson deviance loss on the learning set $\mathcal{D}_1 \cup \mathcal{D}_2$:

$$\theta \;\mapsto\; \mathcal{L}(\theta; \mathcal{D}_1 \cup \mathcal{D}_2) = \frac{2}{|\mathcal{D}_1 \cup \mathcal{D}_2|} \sum_{i \in \mathcal{D}_1 \cup \mathcal{D}_2} e_i \lambda(\boldsymbol{x}_i) - Y_i - Y_i \log\left(e_i \lambda(\boldsymbol{x}_i)\right) + Y_i \log Y_i, \tag{2.3}$$

where $\theta = (\beta_0, (\alpha_r)_r, \gamma_{\texttt{female}}, (\delta_{ag})_{ag}, \beta_1, \beta_2)' \in \mathbb{R}^{11}$ collects all GLM regression parameters. The results show that several region coefficients and age group coefficients are not significant. Including these regions and age groups may lead to over-fitting. We perform the step-wise variable selection on the full model (2.2) in the backward direction given by the Akaike's Information Criterion (AIC). It turns out that we should merge the regions of `Shanghai` and `Others` with the reference region of `Zhejiang`, i.e., only `Hebei` region is significantly different. Moreover, we merge age groups `young_1`, `young_2` and `matured` with the reference age group of `middle_1`. The final Poisson GLM is as follows, under modified covariate space $\mathcal{X} \subset \mathbb{R}^5$,

$$\begin{aligned} \boldsymbol{x}_i \;\mapsto\; \log \lambda(\boldsymbol{x}_i) = \log \lambda(\boldsymbol{x}_i; \theta) = \beta_0 + \alpha_{\texttt{Hebei}} \mathbb{1}_{\texttt{Hebei}}(\texttt{region}_i) + \gamma_{\texttt{female}} \mathbb{1}_{\texttt{female}}(\texttt{gender}_i) + \\ \delta_{\texttt{middle\_2}} \mathbb{1}_{\texttt{middle\_2}}(\texttt{age\_group}_i) + \beta_1 \texttt{car}_i + \beta_2 \texttt{ave\_hours}_i, \end{aligned} \tag{2.4}$$

Table 3 MLE $\widehat{\theta}^{\mathcal{D}_1 \cup \mathcal{D}_2}$ for $\theta \in \mathbb{R}^6$ of model (2.4)

Parameters	Estimate	Standard error	z value	p-value
β_0	−1.7828	0.1243	−14.339	$< 2 \times 10^{-16}$
$\alpha_{\texttt{Hebei}}$	−0.3096	0.1102	−2.810	0.0050
$\gamma_{\texttt{female}}$	0.1585	0.1107	1.431	0.1524
$\delta_{\texttt{middle\_2}}$	−0.4269	0.1674	−2.550	0.0108
β_1	0.1076	0.0301	3.576	0.0003
β_2	0.0470	0.0133	3.548	0.0004

with regression parameter $\theta = (\beta_0, \alpha_{\texttt{Hebei}}, \gamma_{\texttt{female}}, \delta_{\texttt{middle\_2}}, \beta_1, \beta_2)' \in \mathbb{R}^6$. Its maximum likelihood estimator (MLE) $\widehat{\theta}^{\mathcal{D}_1 \cup \mathcal{D}_2}$ on the learning data $\mathcal{D}_1 \cup \mathcal{D}_2$ and the associated p-values are shown in Table 3.

The car drivers in Hebei region have a significantly lower claim frequency than those in Zhejiang, Shanghai and other regions. Female drivers have a slight higher claims frequency than male drivers, but the difference is not statistically significant on a 5% level. Drivers at ages [40, 44] have a lower claims frequency than those at other ages. For cars older than 3 years a log-linear functional form is supported, and the claim frequency is increasing log-linearly with the average driving hours per week.

2.2.2 Out-of-sample test error

The predictive performance of the estimated claim frequency model (2.4) is evaluated by the out-of-sample Poisson deviance loss on the test data $\mathcal{D}_3$ (called test error):

$$\mathcal{L}(\widehat{\theta}^{\mathcal{D}_1 \cup \mathcal{D}_2}; \mathcal{D}_3) = \frac{2}{|\mathcal{D}_3|} \sum_{i \in \mathcal{D}_3} e_i \lambda(\boldsymbol{x}_i; \widehat{\theta}^{\mathcal{D}_1 \cup \mathcal{D}_2}) - Y_i - Y_i \log \left(e_i \lambda(\boldsymbol{x}_i; \widehat{\theta}^{\mathcal{D}_1 \cup \mathcal{D}_2}) \right) + Y_i \log Y_i, \tag{2.5}$$

where $\widehat{\theta}^{\mathcal{D}_1 \cup \mathcal{D}_2}$ denotes the MLE based on learning set $\mathcal{D}_1 \cup \mathcal{D}_2$. Preference is given to the model with the smaller test error. The test error of model (2.4) is 1.1230 and the in-sample learning error received through (2.3) is 1.0205. For comparison, we establish the homogeneous model

$$\lambda(\cdot) \equiv \bar{\lambda}^{\mathcal{D}_1 \cup \mathcal{D}_2} = \frac{\sum_{i \in \mathcal{D}_1 \cup \mathcal{D}_2} Y_i}{\sum_{i \in \mathcal{D}_1 \cup \mathcal{D}_2} e_i}. \tag{2.6}$$

The test error of the homogeneous model (without any covariates and systematic effects) is 1.1703 and the learning error is 1.0717, thus, clearly bigger than in model (2.4). The latter model is our benchmark for all subsequent derivations.

3 Claims frequency modeling using telematics data

We establish a first predictive model that is based on telematics car driving data, only. The portfolio used is identical to the one of Sect.2, we also refer to Table 2. Each driver $i = 1, \ldots, n$ is described by a so-called speed-acceleration (v-a) heatmap $\boldsymbol{Z}_i$, and its explicit

construction from telematics car driving data is described in detail in Sect.3.2. The general Poisson regression setting is given as follows

$$Y_i \overset{\text{ind.}}{\sim} \text{Poisson}(e_i \widetilde{\lambda}_i \rho(\boldsymbol{Z}_i)), \qquad \text{for } i = 1, \dots, n, \tag{3.1}$$

where $e_i > 0$ is the exposure of car driver i, $\widetilde{\lambda}_i > 0$ is a given prior estimate of claim frequency for car driver i, and $\boldsymbol{Z}_i \mapsto \rho(\boldsymbol{Z}_i) > 0$ is a telematics *driving behavior risk factor*. In Sect. 3.2, we construct the *v-a* heatmap $\boldsymbol{Z}_i$; in Section 3.3, we apply a densely connected feed-forward neural network to estimate the driving behavior risk factor $\rho(\boldsymbol{Z}_i)$; and in Sect. 3.4, we challenge the densely connected feed-forward neural network by a convolutional neural network.

3.1 Telematics car driving data

We discuss our telematics data in this section. Our telematics data is collected from three independent sensors: GPS signal, instrumental panel and three-axis accelerometer. The GPS location is transmitted second by second, and from these GPS locations we can calculate average speed and acceleration every second. The instrumental panel provides the speed of the car every second as shown to the driver, and from this we can also calculate the average acceleration every second. Finally, the accelerometer records acceleration in all three directions. Unfortunately, the precision of these measurements may be poor because these devices need to be recalibrated regularly, and from similar telematics projects it is known that this recalibration might cause some difficulties, for instance, it might be influenced by the fact whether a car is parked at a steep road or in a flat plane during recalibration. We also observe difficulties in our data with the accelerometer device, i.e., the GPS signal and the instrumental panel being in line, but the accelerometer showing different measurements. For this analysis we have decided to rely on the information from the instrumental panel because it sometimes happens that the GPS signal gets lost (or is imprecise) when, for instance, driving through a tunnel.

3.2 Speed-acceleration heatmaps

We compress individual telematics car driving information into a so-called speed-acceleration (*v-a*) heatmap for each driver i. These *v-a* heatmaps describe how drivers accelerate and brake at different speeds. As highlighted in Gao and Wüthrich (2019), telematics car driving data easily results in big data of several TBs. This makes it necessary to compress this data appropriately to make it useful for statistical modeling. This data compression is performed as in Gao et al. (2019), basically, telematics data is aggregated in a suitable way. In Gao et al. (2019) we have studied the speed of convergence of such aggregations and we have seen that it takes roughly three months of data until the aggregation has converged. For this reason, all subsequent analysis will be based on the three months of driving experience from 01/05/2016 to 31/07/2016. This is the time period when we have a maximal number of cars with telematics data. Remark that we did not find seasonality in the heatmap constructions, and even if there was, we judge all drivers on common ground because we choose the identical time period.

Denote the *v-a* rectangle by $R = (0, 80]\text{km/h} \times [-2, 2]\text{m/s}^2$. Speed is truncated within (0, 80]km/h since we want to remove the idle phase and there are not sufficiently many observations above 80km/h to receive stable heatmaps. Acceleration is capped within

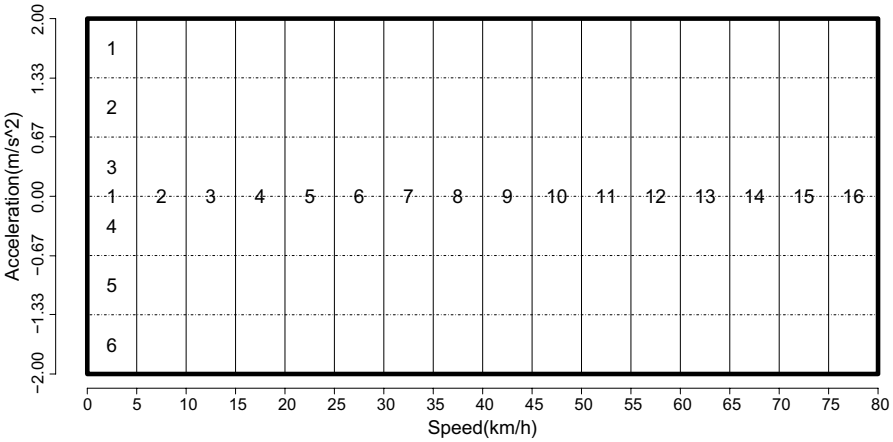

Fig. 3 Partition of $R = (0, 80]\text{km/h} \times [-2, 2]\text{m/s}^2$ in equally spaced sub-rectangles

$[-2, 2]\text{m/s}^2$ since there are not sufficiently many observation outside of this interval. Note that these accelerations and decelerations are moderate, Weidner et al. (2016, 2017) and Sun et al. (2020) work with bigger values.

The acceleration interval $[-2, 2]\text{m/s}^2$ is divided into 6 equally spaced sub-intervals $j = 1, \ldots, 6$, and the speed interval (0, 80]km/h is divided into 16 equally spaced sub-intervals $k = 1, \ldots, 16$; see Fig. 3. For each speed sub-interval $k = 1, \ldots, 16$, the *acceleration pattern* of driver i is defined as the (probability) distribution of accelerations in that speed sub-interval:

$$z_{i,j,k} = \frac{t_{i,j,k}}{\sum_{j=1}^{6} t_{i,j,k}} \geq 0, \tag{3.2}$$

where $t_{i,j,k}$ is the total time spent in acceleration sub-interval j for given speed sub-interval k of driver i. We have normalization $\sum_{j=1}^{6} z_{i,j,k} = 1$, thus, for every k we receive probabilities of a categorical distribution.

The driving behavior of every car driver i is represented by a 6×16 matrix, called *v-a heatmap*,

$$\mathbf{Z}_i = (z_{i,j,k})_{j=1:6,k=1:16} \in [0, 1]^{6\times 16}. \tag{3.3}$$

We plot the *v-a* heatmaps of the two selected drivers 44 and 191 (belonging to the test data $\mathcal{D}_3$) in Fig. 4. It shows that driver 191 accelerates and brakes much more intensely than driver 44. In Sect. 4, we show that driver 191 has the largest driving behavior risk factor and driver 44 has the smallest one in our test data $\mathcal{D}_3$, i.e., these are the two extreme cases in $\mathcal{D}_3$.

It is standard in image recognition that inputs to convolutional neural networks are three-dimensional arrays consisting of height×width×channels. Therefore, we transform (by a slight abuse of notation) the *v-a* heatmap to a three-dimensional array $\mathbf{Z}_i \in [0, 1]^{6\times 16\times 1}$.

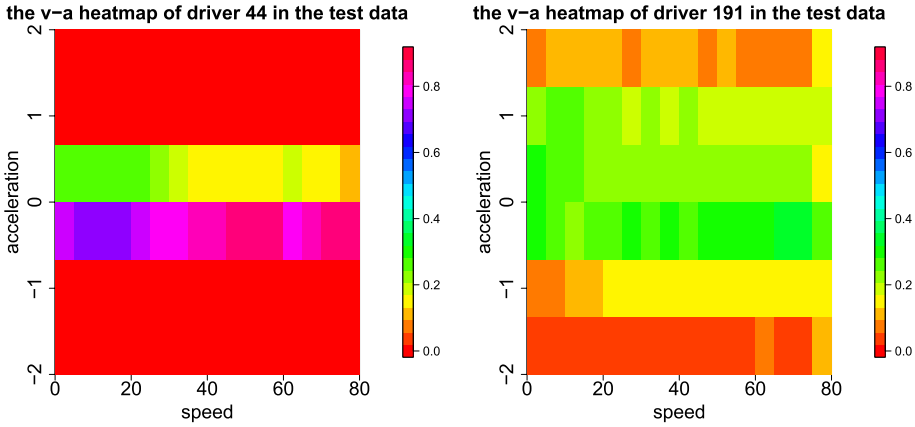

Fig. 4 *v-a* heatmaps of drivers 44 and 191 belonging to the test data $\mathcal{D}_3$

3.3 A densely connected neural network for claim frequency prediction

The heatmap $\boldsymbol{Z}_i$ has dimension of $6 \times 16 \times 1$, and usually we need feature engineering of $\boldsymbol{Z}_i$ before using it in a claim frequency model, this is the approach taken in Gao et al. (2019). Instead of manually feature engineering in a two-step modeling approach, we apply a densely connected neural network to do both feature engineering and regression modeling simultaneously.

3.3.1 Densely connected feed-forward neural network architecture

We design a densely connected feed-forward neural network with two hidden layers. The `Keras` code is provided in "Appendix A", and the architecture is shown in Listing 1. We choose the number of neurons in each of the two hidden layers as $m_1 = 30$ and $m_2 = 10$. Each row of Listing 1 shows the layer name, the type of layer (in the bracket), the dimension of the layer, the number of parameters and the preceding layer name(s) it is connected to.

Listing 1: Architecture of the densely connected feed-forward neural network (dnn).

```
_________________________________________________________________________________
Layer (type)                 Output Shape              Param     Connected to
=================================================================================
heatmap (InputLayer)         [(None, 6, 16, 1)] 0
_________________________________________________________________________________
heatmap_flat (Flatten)       (None, 96)                0         heatmap[0][0]
_________________________________________________________________________________
heatmap_dense1 (Dense)       (None, 30)                2910      heatmap_flat[0][0]
_________________________________________________________________________________
heatmap_drop1 (Dropout)      (None, 30)                0         heatmap_dense1[0][0]
_________________________________________________________________________________
heatmap_dense2 (Dense)       (None, 10)                310       heatmap_drop1[0][0]
_________________________________________________________________________________
heatmap_drop2 (Dropout)      (None, 10)                0         heatmap_dense2[0][0]
_________________________________________________________________________________
heatmap_factor (Dense)       (None, 1)                 11        heatmap_drop2[0][0]
_________________________________________________________________________________
vol (InputLayer)             [(None, 1)]               0
_________________________________________________________________________________
response (Multiply)          (None, 1)                 0         heatmap_factor[0][0]
                                                                 vol[0][0]
=================================================================================
Total params: 3,231
Trainable params: 3,231
Non-trainable params: 0
_________________________________________________________________________________
```

We have two input layers, one flatten layer, three dense layers, two dropout layers and one multiply layer. More specifically, the following layers are connected sequentially to form the densely connected feed-forward neural network denoted by dnn.

The flatten layer Each element of $\boldsymbol{Z}_i$ is on unit scale [0, 1], therefore, we input $\boldsymbol{Z}_i$ directly to the neural network without any pre-processing. We flatten the array $\boldsymbol{Z}_i$ to a $m_0 = 6 \cdot 16 = 96$ dimensional vector $\boldsymbol{z}_i^1 = (z_{i,j}^1)_{j=1:m_0}$ through the flatten layer heatmap_flat:

$$\phi^1 : [0,1]^{6\times 16\times 1} \to [0,1]^{m_0}, \qquad \boldsymbol{Z}_i \mapsto \phi^1(\boldsymbol{Z}_i) = \boldsymbol{z}_i^1 = (z_{i,1}^1, \ldots, z_{i,m_0}^1)'.$$

This layer discards the spatial structure. There is no parameter in this layer involved.

The first dense layer The m_0-dimensional vector $\boldsymbol{z}_i^1$ is projected to a m_1-dimensional space through the first dense hidden layer heatmap_dense1:

$$\phi^2 : [0,1]^{m_0} \to (-1,1)^{m_1}, \qquad \boldsymbol{z}_i^1 \mapsto \phi^2(\boldsymbol{z}_i^1) = \boldsymbol{z}_i^2 = (z_{i,1}^2, \ldots, z_{i,m_1}^2)', \tag{3.4}$$

where

$$z_{i,j}^2 = \tanh\left(\theta_{0,j}^2 + \sum_{l=1}^{m_0} \theta_{l,j}^2 z_{i,l}^1\right), \quad \text{for} \quad j = 1, \ldots, m_1.$$

We use the hyperbolic tangent activation; other activation functions lead to similar results. There are $(m_0 + 1)m_1$ parameters $\theta_{l,j}^2 \in \mathbb{R}$ in this layer.

The first dropout layer A dropout layer heatmap_drop1 with dropout rate d_1 is inserted between the first dense layer and the second dense layer. The dropout layer does

not affect the neural network architecture, and it does not involve any model parameters. The dropout layer only affects the calibration, in particular, it prevents from over-fitting because in each gradient descent step neurons are removed independently and randomly with the given dropout rate d_1. We tune the dropout rate $d_1 = 0.1$ according to the validation error; see model calibration in Sect. 3.3.2.

The second dense layer The m_1-dimensional vector z_i^2 is projected to a m_2-dimensional space through the second dense layer heatmap_dense2:

$$\phi^3 : (-1,1)^{m_1} \to (-1,1)^{m_2}, \quad z_i^2 \mapsto \phi^3(z_i^2) = z_i^3 = (z_{i,1}^3, \ldots, z_{i,m_2}^3)' \tag{3.5}$$

where

$$z_{i,j}^3 = \tanh\left(\theta_{0,j}^3 + \sum_{l=1}^{m_1} \theta_{l,j}^3 z_{i,l}^2\right), \quad \text{for} \quad j = 1, \ldots, m_2.$$

There are $(m_1 + 1)m_2$ parameters $\theta_{l,j}^3 \in \mathbb{R}$ in this layer. We tune the number of neurons $m_2 = 10$.

The second dropout layer A dropout layer heatmap_drop2 with dropping rate d_2 is inserted between the second dense layer and the third dense layer. The explanation is similar to the first dropout layer. We tune the dropout rate $d_2 = 0.1$.

The third dense layer The m_2-dimensional vector z_i^3 is projected to a 1-dimensional space through the third dense layer heatmap_factor:

$$\phi^4 : (-1,1)^{m_2} \to \mathbb{R}_+, \quad z_i^3 \mapsto \phi^4(z_i^3) = z_i^4 = \exp\left(\theta_0^4 + \sum_{j=1}^{m_2} \theta_j^4 z_{i,j}^3\right). \tag{3.6}$$

We call z_i^4 the *driving behavior risk factor* of driver i. We use the exponential activation function since the driving behavior risk factor must be positive, see (3.1), and because the log-link is the canonical link in the Poisson GLM. There are $m_2 + 1$ parameters $\theta_j^4 \in \mathbb{R}$ in this layer. Altogether, the above 6 layers define a mapping (composition) from the heatmap to the driving behavior risk factor in (3.1):

$$\rho^{\text{dnn}} : [0,1]^{6\times 16\times 1} \to \mathbb{R}_+, \quad \boldsymbol{Z}_i \mapsto \rho^{\text{dnn}}(\boldsymbol{Z}_i) = \left(\phi^4 \circ \phi^3 \circ \phi^2 \circ \phi^1\right)(\boldsymbol{Z}_i) = z_i^4. \tag{3.7}$$

The multiply layer Finally, we multiply z_i^4 with exposure $e_i > 0$ and prior claim frequency estimate $\widetilde{\lambda}_i$ to get the output of the neural network (layer response) $e_i \widetilde{\lambda}_i z_i^4 = e_i \widetilde{\lambda}_i \rho^{\text{dnn}}(\boldsymbol{Z}_i)$. Note that the term $e_i \widetilde{\lambda}_i$ goes into the neural network through the input layer vol, and it acts as an offset in the regression model.

3.3.2 Densely connected feed-forward neural network model calibration

We choose the average claim frequency from the homogeneous model (2.6) as prior estimates, i.e., $\widetilde{\lambda}_i = \bar{\lambda}^{\mathcal{D}_1 \cup \mathcal{D}_2}$. Denote the vector of all neural network parameters by θ. We apply the `adam` version of the gradient decent method to iteratively find a good parameter estimate for θ. As objective function we choose the Poisson deviance loss having in-sample training error on $\mathcal{D}_1$:

$$\mathcal{L}(\theta; \mathcal{D}_1) = \frac{2}{|\mathcal{D}_1|} \sum_{i \in \mathcal{D}_1} e_i \widetilde{\lambda}_i \rho^{\text{dnn}}(\boldsymbol{Z}_i; \theta) - Y_i - Y_i \log\left(e_i \widetilde{\lambda}_i \rho^{\text{dnn}}(\boldsymbol{Z}_i; \theta)\right) + Y_i \log Y_i. \tag{3.8}$$

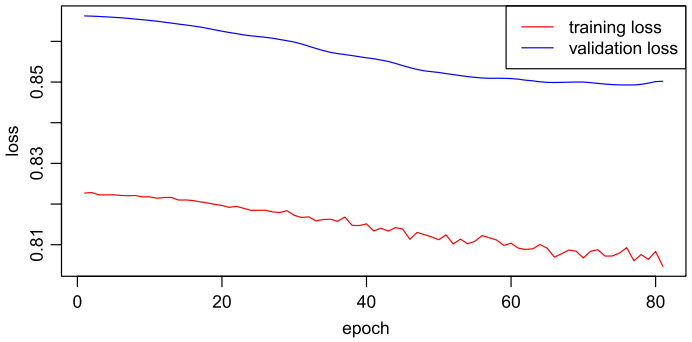

Fig. 5 Steepest gradient descent calibration of the densely connected neural network (dnn)

Table 4 Learning error and test error of the models studied

Error	Homogeneous (2.6)	GLM (2.4)	dnn Listing 1	cnn Listing 2	dnn + glm (4.1)	cnn + glm (4.2)
Learning error	1.0717	1.0205	1.0376	1.0415	0.9982	0.9992
Test error	1.1703	1.1230	1.1035	1.1075	1.0655	1.0690
Reduction in test error		0.0473	0.0668	0.0628	0.1048	0.1013

We emphasize that our training data set $\mathcal{D}_1$ is small. Therefore, we perform steepest gradient descent on $\mathcal{D}_1$, and not any stochastic gradient descent method.

Figure 5 shows the steepest gradient descent performance $\theta \mapsto \mathcal{L}(\theta;\mathcal{D}_1)$ on the training data $\mathcal{D}_1$ (red color) and the corresponding out-of-sample validation losses $\theta \mapsto \mathcal{L}(\theta;\mathcal{D}_2)$ on the validation data $\mathcal{D}_2$.[1] The model starts to over-fit after roughly 80 iterations (note that the training error is not monotone decreasing due to the presence of dropouts). Using the callback_early_stopping option we retrieve the parameters with the lowest validation loss after not improving in a sequence of 5 iterations. It takes 4 seconds to complete calibration on a 1.3 GHz dual Intel Core i5 computer. Note that the initial values of the parameters in the third dense layer heatmap_factor are set to 0; see `Keras` code in "Appendix A". Hence, gradient descent calibration of the neural network starts from the homogeneous model $e_i\bar{\lambda}^{\mathcal{D}_1\cup\mathcal{D}_2}$. The hyper-parameters m_1, m_2, d_1, d_2 are selected as 30, 10, 0.1, 0.1 by monitoring the validation error.

Table 4 provides the results in column 'dnn'. The densely connected feed-forward neural network with driving behavior risk factor $\rho^{\text{dnn}}(\boldsymbol{Z}_i)$ outperforms the GLM out-of-sample and we observe a test error decrease from 1.1230 to 1.1035. Thus, for this data partition the *v-a* heatmap $\boldsymbol{Z}_i$ has better predictive power than the 5 classical risk factors $\boldsymbol{x}_i$ (in our GLM).

[1] Note that internally `Keras` drops all (constant) terms that are not relevant for the gradient descent algorithm. For this reason, the *y*-axis of Fig. 5 does not match Poisson deviance losses because the constant terms $\sum_{i\in\mathcal{D}_1}\left(Y_i \log Y_i - Y_i\right)/|\mathcal{D}_1|$ and correspondingly for $\mathcal{D}_2$ are missing, and the scaling factor of 2 is not considered.

3.4 Convolutional neural network for claim frequency prediction

It is natural to replace the densely connected feed-forward neural network to process the *v-a* heatmap $\boldsymbol{Z}_i$ by a convolutional neural network. In the former approach we apply a flatten layer ϕ^1 in the first step, this implies that we lose the topological structure of the *v-a* heatmap. On the other hand, convolutional neural networks are designed to handle spatial patterns.

3.4.1 The architecture

We design a convolutional neural network with two convolution layers. The first convolution layer has a 6×1 convolution window with q filters and stride 1. The second convolution layer has a 1×2 convolution window with 1 filter and stride 2. These two convolution layers are motivated by fact that they allow for interpretation of our results, see the following description of these two convolution layers and their interpretation in Sect. 4.2. The `Keras` code is provided in "Appendix A"; the architecture is shown in Listing 2, where we choose $q = 2$.

Listing 2: Architecture of the convolutional neural network (cnn).

```
_________________________________________________________________________________
Layer (type)                 Output Shape              Param     Connected to
=================================================================================
heatmap (InputLayer)         [(None, 6, 16, 1)] 0
_________________________________________________________________________________
heatmap_conv1 (Conv2D)       (None, 1, 16, 2)    14        heatmap[0][0]
_________________________________________________________________________________
heatmap_conv2 (Conv2D)       (None, 1, 8, 1)     5         heatmap_conv1[0][0]
_________________________________________________________________________________
heatmap_flat (Flatten)       (None, 8)           0         heatmap_conv2[0][0]
_________________________________________________________________________________
heatmap_factor (Dense)       (None, 1)           9         heatmap_flat[0][0]
_________________________________________________________________________________
vol (InputLayer)             [(None, 1)]         0
_________________________________________________________________________________
response (Multiply)          (None, 1)           0         heatmap_factor[0][0]
                                                           vol[0][0]
=================================================================================
Total params: 28
Trainable params: 28
Non-trainable params: 0
_________________________________________________________________________________
```

An obvious difference between Listings 1 and 2 is the number of parameters. The convolutional neural network has 28 parameters while the densely connected feed-forward neural network has 3, 231 parameters. This is because the convolution windows are small and their parameters are shared in different locations of the heatmap. For this reason, we do not use dropout layers here. We have two input layers, two convolution layers, one flatten layer, one dense layer and one multiply layer. More specifically, the following layers are connected sequentially to form the convolutional neural network.

The first convolution layer We let a 6×1 convolution window moving along the speed direction of the *v-a* heatmap with stride 1 to extract q acceleration patterns in each speed

sub-interval $(5(k-1), 5k]$ km/h for $k = 1, \ldots, 16$; see Fig. 3. The layer heatmap_conv1 defines the following mapping:

$$\psi^1 : [0,1]^{6\times 16\times 1} \to (-1,1)^{1\times 16\times q}, \quad \boldsymbol{Z}_i \mapsto \psi^1(\boldsymbol{Z}_i) = \boldsymbol{Z}_i^1 = (z_{i,j,k,l}^1)_{j=1,k=1:16,l=1:q}, \tag{3.9}$$

where

$$z_{i,j,k,l}^1 = \tanh\left(\theta_{0,l}^1 + \sum_{t=1}^{6} \theta_{t,l}^1 z_{i,t,k,1}\right), \qquad \text{for} \quad j = 1, k = 1, \ldots, 16, l = 1, \ldots, q. \tag{3.10}$$

We call $z_{i,j,k,l}^1$ as the l-th acceleration pattern in the speed sub-interval $(5(k-1), 5k]$ km/h for driver i, extracted by the first convolution layer. There are $7q$ parameters $\theta_{t,l}^1 \in \mathbb{R}$ in this layer. Note that we use the same symbols for activations z and parameters θ as in the densely connected feed-forward neural network. When necessary, we will clarify to avoid confusion.

The second convolution layer A 1×2 convolution window is moving along the horizontal direction of the v-a heatmap with stride 2 to extract one acceleration pattern in the speed sub-interval $(10(k-1), 10k]$ km/h for $k = 1, \ldots, 8$; see Fig. 3. The layer heatmap_conv2 defines the following mapping:

$$\psi^2 : (-1,1)^{1\times 16\times q} \to (-1,1)^{1\times 8\times 1}, \quad \boldsymbol{Z}_i^1 \mapsto \psi^2(\boldsymbol{Z}_i^1) = \boldsymbol{Z}_i^2 = (z_{i,j,k,l}^2)_{j=1,k=1:8,l=1}, \tag{3.11}$$

where

$$z_{i,j,k,l}^2 = \tanh\left(\theta_0^2 + \sum_{s=1}^{q} \theta_{s,1}^2 z_{i,j,2k-1,s}^1 + \theta_{s,2}^2 z_{i,j,2k,s}^1\right), \text{ for } \quad j = 1, k = 1, \ldots, 8, l = 1. \tag{3.12}$$

We call $z_{i,j,k,l}^2$ as the acceleration pattern in the speed sub-interval $(10(k-1), 10k]$ km/h for driver i, extracted by the second convolution layer. There are $2(q+1)$ parameters $\theta_{s,t}^2 \in \mathbb{R}$ in this layer. One may increase the filter numbers of this layer and add more convolution layers behind this layer, but they always provide a similar predictive performance.

The flatten layer We flatten the array $\boldsymbol{Z}^2$ into a 8-dimensional vector through the layer heatmap_flat:

$$\psi^3 : (-1,1)^{1\times 8\times 1} \to (-1,1)^8, \quad \boldsymbol{Z}_i^2 \mapsto \psi^3(\boldsymbol{Z}_i^2) = \boldsymbol{z}_i^3 = (z_{i,1}^3, \ldots, z_{i,8}^3)', \tag{3.13}$$

where

$$z_{i,k}^3 = z_{i,j,k,l}^2 = z_{i,1,k,1}^2, \quad \text{for} \quad k = 1, \ldots, 8.$$

There is no parameter in the flatten layer. The purpose of this layer is to transform the array into a vector so it can be used as the input of the dense layer followed.

The dense layer The 8-dimensional vector $\boldsymbol{z}^3$ is projected to a 1-dimensional space through the dense layer heatmap_factor:

$$\psi^4 : (-1,1)^8 \to \mathbb{R}_+, \quad z_i^3 \mapsto \psi^4(z_i^3) = z_i^4 = \exp\left(\theta_0^4 + \sum_{k=1}^{8} \theta_k^4 z_{i,k}^3\right). \tag{3.14}$$

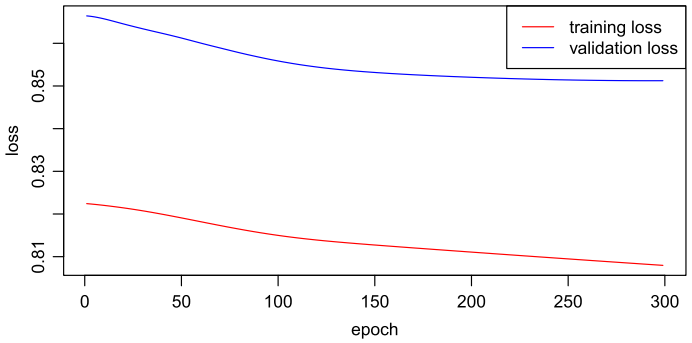

Fig. 6 Steepest gradient descent calibration of the convolutional neural network (cnn)

We call z_i^4 the *driving behavior risk factor* of driver i. There are 9 parameters θ_k^4 in this layer. For the same reasons as above, we use the exponential activation function. Altogether, the above 4 layers define the mapping from the heatmaps to the driving behavior risk factor in (3.1):

$$\rho^{\text{cnn}} : [0,1]^{6\times 16\times 1} \to \mathbb{R}_+, \quad \boldsymbol{Z}_i \mapsto \rho^{\text{cnn}}(\boldsymbol{Z}_i) = \left(\psi^4 \circ \psi^3 \circ \psi^2 \circ \psi^1\right)(\boldsymbol{Z}_i) = z_i^4 \tag{3.15}$$

The multiply layer Finally we multiply z_i^4 with the exposure $e_i > 0$ and the prior claims frequency estimate $\widetilde{\lambda}_i$ to get the output of the neural network (layer response) $e_i \widetilde{\lambda}_i z_i^4 = e_i \widetilde{\lambda}_i \rho^{\text{cnn}}(\boldsymbol{Z}_i)$.

3.4.2 Convolutional neural network model calibration

The calibration of the convolutional neural network is similar to the calibration of the densely connected feed-forward neural network in Sect. 3.3.2. Again, we choose the average claims frequency of the homogeneous model (2.6) as the prior estimates. The initial values of the parameters in the dense layer are set to 0; see `keras` code in "Appendix A". So we start calibrating from the homogeneous model $e_i \bar{\lambda}^{\mathcal{D}_1 \cup \mathcal{D}_2}$. The algorithm converges fast as shown in Fig. 6, it only takes 12 seconds. The hyper-parameter q is selected as 2 by monitoring the validation error.

The results are given in Table 4. Using the convolutional neural network with driving behavior risk factors, we improve the test error of the GLM (1.1075 versus 1.1230), but the out-of-sample performance is slightly worse compared to the densely connected network. However, we have a slight preference for the convolutional network approach because it keeps the number of parameters involved on a small scale.

4 Boosting classical risk factors with telematics data

Equation (3.1) introduces a way of incorporating classical risk factors into neural networks. We choose the estimated claims frequency from GLM (2.4) as prior claim frequency estimates, i.e., $\widetilde{\lambda}_i = \lambda(\boldsymbol{x}_i; \widehat{\theta}) = \widehat{\lambda}(\boldsymbol{x}_i)$. This is in the spirit of the combined actuarial neural network (CANN) approach proposed in Wüthrich and Merz (2019), which in our context boosts the GLM frequencies $\widehat{\lambda}(\boldsymbol{x}_i)$ with telematics driving risk factors $\rho(\boldsymbol{Z}_i)$.

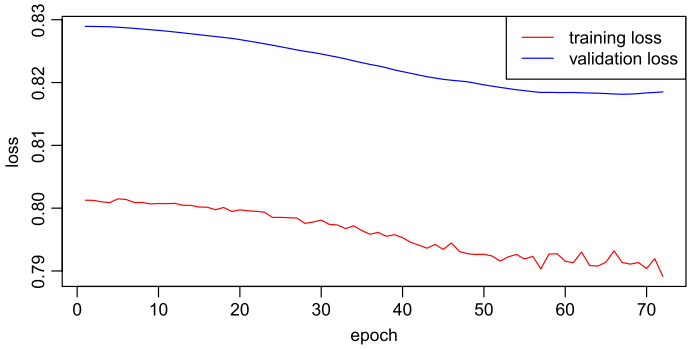

Fig. 7 Calibration of densely connected neural network boosted Poisson GLM (dnn + glm)

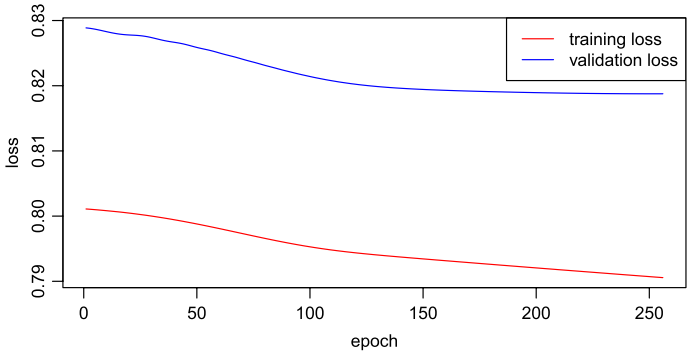

Fig. 8 Calibration of convolutional neural network boosted Poisson GLM (cnn + glm)

4.1 Telematics neural network boosted Poisson generalized linear model

CANN boosts the Poisson GLM (2.4) with either the densely connected neural network of Listing 1 (dnn) or the convolutional neural network of Listing 2 (cnn): For $i = 1, \dots, n$ we assume

$$Y_i \overset{\text{ind.}}{\sim} \text{Poisson}(e_i \widehat{\lambda}(\boldsymbol{x}_i) \rho^{\text{dnn}}(\boldsymbol{Z}_i)), \tag{4.1}$$

$$Y_i \overset{\text{ind.}}{\sim} \text{Poisson}(e_i \widehat{\lambda}(\boldsymbol{x}_i) \rho^{\text{cnn}}(\boldsymbol{Z}_i)), \tag{4.2}$$

where $e_i \widehat{\lambda}(\boldsymbol{x}_i)$ is the GLM estimated expected number of claims of driver i with classical risk factors $\boldsymbol{x}_i$, see (2.4). The architectures of the networks in (4.1) and (4.2) are chosen identical to the ones in Sects.3.3.2 and 3.4.2. Also the network calibration goes along the same lines as above, the gradient descent results are shown in Figs. 7 and 8.

Note that both gradient descent calibrations start from the Poisson GLM (2.4) by the way we initialize the algorithm: the starting points in Figs. 7 and 8 are below those in Figs. 5 and 6 because we already start from a reasonably good GLM, and the convergence rates in Figs. 7 and 8 are very fast. Note that we keep the GLM parameters $\hat{\theta}$ of the classical covariates during the neural network calibration for two reasons: firstly, this keeps the

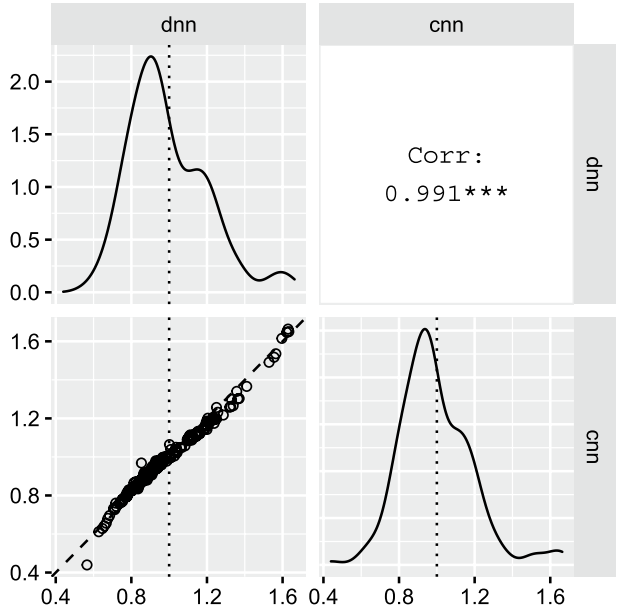

Fig. 9 Comparison of driving behavior risk factors $\rho^{\text{dnn}}(\boldsymbol{Z}_i)$ and $\rho^{\text{cnn}}(\boldsymbol{Z}_i)$ on test data $i \in \mathcal{D}_3$

interpretation of the GLM prediction and gives a stable neural network calibration; secondly, the interpretation of ρ is intuitively obtained as a modification of the GLM prediction $\hat{\lambda}$. In this sense, we boost the GLM by integrating the GLM prediction as an offset into the networks. If more data would be available one could think of also further training $\hat{\theta}$, this would allow for more general interactions between the classical covariates and telematics information.

4.1.1 Comparison of driving behavior risk factors

We start by comparing the driving behavior risk factors $\rho^{\text{dnn}}(\boldsymbol{Z}_i)$ and $\rho^{\text{cnn}}(\boldsymbol{Z}_i)$ on the test data $i \in \mathcal{D}_3$ in Fig. 9.

From the graphs in Fig. 9 we conclude that both networks provide almost identical driving risk factors $\rho^{\text{dnn}}(\boldsymbol{Z}_i) \approx \rho^{\text{cnn}}(\boldsymbol{Z}_i)$ in (0.4, 1.6). In fact, the more complex densely connected feed-forward neural network can be replaced by a simpler convolutional neural network having only 28 parameters. In Fig. 4, we show the *v-a* heatmaps with the minimal and maximal driving behavior risk factors, i.e., high acceleration and braking obviously triggers a higher frequency in our example.

4.1.2 Comparison of different models

Table 4 and Fig. 10 (left) compare the different models. The *v-a* heatmap boosted Poisson GLMs (dnn + glm and cnn + glm) have the best out-of-sample predictive performance. Thus, we conclude that *v-a* heatmaps $\boldsymbol{Z}_i$ contain information beyond classical actuarial covariates $\boldsymbol{x}_i$, and on the other hand these *v-a* heatmaps $\boldsymbol{Z}_i$ do not fully replace classical actuarial covariates $\boldsymbol{x}_i$. The former statement is clear because we believe that *v-a* heatmaps $\boldsymbol{Z}_i$ best describe driving styles. However, also the latter makes sense because *v-a* heatmaps $\boldsymbol{Z}_i$ will interact with road conditions, car type, etc., which may be reflected in `region`,

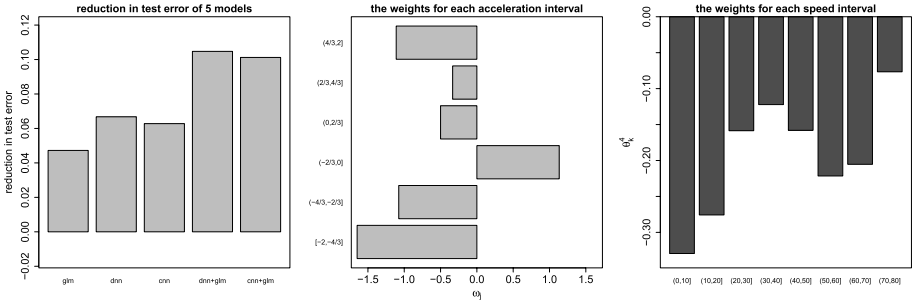

Fig. 10 (left) Reduction in test error of all models studied (compared to the homogeneous model); (middle) weights ω_j for each acceleration sub-interval; (right) weights θ_k^4 for each speed sub-interval

`car` and other classical actuarial covariates. That is, classical actuarial covariates may indicate under which circumstances the telematics data has been collected.

4.2 Interpretation of the convolutional neural network results

We explain how the acceleration pattern $z_{i,k}^3 = z_{i,1,k,1}^2$ in the speed sub-interval $(10(k-1), 10k]$ km/h is constructed. In "Appendix B", we approximate $z_{i,k}^3$ by a linear combination of $z_{i,j,2k-1,1}^0$ and $z_{i,j,2k,1}^0$, namely,

$$z_{i,k}^3 \;\approx\; \omega_0 + \sum_{j=1}^{6} \omega_j \frac{z_{i,j,2k-1,1} + z_{i,j,2k,1}}{2}, \qquad \text{for } k = 1, \ldots, 8, \tag{4.3}$$

where $\omega_j = \theta_{1,1}^2 \theta_{j,1}^1 + \theta_{2,1}^2 \theta_{j,2}^1 + \theta_{1,2}^2 \theta_{j,1}^1 + \theta_{2,2}^2 \theta_{j,2}^1$ is interpreted as the weight of each acceleration sub-interval $(2 - 2j/3, 2 - 2(j-1)/3]$ m/s^2, $j = 1, \ldots, 6$. We plot these weights ω_j in Fig. 10 (middle).

The signs of the weights ω_j for hard accelerating and hard braking are opposite to those for smooth driving. We interpret $z_{i,k}^3$ as the relative frequency of smooth driving in the speed sub-interval $(10(k-1), 10k]$ km/h for $k = 1, \ldots, 8$. The absolute values of the weights for hard braking are larger than those for hard accelerating. It seems that hard braking plays a more important role than hard accelerating in the acceleration pattern $z_{i,k}^3$.

We draw the pair plots of $z_{i,k}^3$ on the test data $i \in \mathcal{D}_3$ in Fig. 11. It shows that acceleration patterns $z_{i,k}^3$ in neighboring speed intervals are quite similar, with high correlations of around 0.9. Acceleration patterns $z_{i,k}^3$ in (10, 40] km/h tend be smaller than those at other speeds, indicating that drivers tend to hard accelerate and brake in (10, 40] km/h more often than in other speed intervals (which, of course, makes perfect sense).

Acceleration patterns $z_{i,k}^3$ in different speed intervals are combined to obtain the *driving behavior risk factor* through the (last) dense layer, see (3.14). We interpret the parameters $(\theta_k^4)_{k=1:8}$ in this last dense layer as the weights for each speed sub-interval $(10(k-1), 10k]$ km/h for $k = 1, \ldots, 8$. We plot $(\theta_k^4)_{k=1:8}$ in Fig. 10 (right). The absolute values of θ_k^4 are decreasing with speeds. It seems that the acceleration pattern $z_{i,k}^3$ in low speeds plays a more important role in constructing the (overall) driving behavior risk factor than at high speeds. Of course, also this makes sense because frequent claim counts often happen at low speeds, say, in urban area.

Table 5 Training-validation-test partitions; partition 0 is the split used in the previous sections

Partitions	Training set (60%)	Validation set (20%)	Test set (20%)
0	$\mathcal{D}_1 = \mathcal{D}_{1,1} \cup \mathcal{D}_{1,2} \cup \mathcal{D}_{1,3}$	$\mathcal{D}_2$	$\mathcal{D}_3$
1	$\mathcal{D}_1 = \mathcal{D}_{1,1} \cup \mathcal{D}_{1,2} \cup \mathcal{D}_{1,3}$	$\mathcal{D}_3$	$\mathcal{D}_2$
2	$\mathcal{D}_{1,1} \cup \mathcal{D}_2 \cup \mathcal{D}_3$	$\mathcal{D}_{1,2}$	$\mathcal{D}_{1,3}$
3	$\mathcal{D}_{1,2} \cup \mathcal{D}_2 \cup \mathcal{D}_3$	$\mathcal{D}_{1,3}$	$\mathcal{D}_{1,1}$
4	$\mathcal{D}_{1,3} \cup \mathcal{D}_2 \cup \mathcal{D}_3$	$\mathcal{D}_{1,1}$	$\mathcal{D}_{1,2}$

4.3 Sensitivity test

We have a comparably small portfolio of $n = 973$ car drivers. It is necessary to conduct a sensitivity test to see whether the above conclusions are still valid for different training-validation-test partitions. We further partition the training set $\mathcal{D}_1$ into 3 disjoint data sets of approximately the same size $\mathcal{D}_{1,1}, \mathcal{D}_{1,2}, \mathcal{D}_{1,3}$, and design four training-validation-test partitions as shown in Table 5. Note that partition 0 is the data split used in the previous sections.

For each training-validation-test partition, we fit all six models of Table 4. For each model, we calculate the reduction in test error compared to the homogeneous model. For the convolutional neural network boosted GLM (4.2), we calculate the weights for each acceleration sub-interval and for each speed sub-interval. All the results are shown in Fig. 12, and they should be compared to Fig. 10.

Similar reduction in test errors over all partitions of Table 5 for both neural networks reconfirm that both architectures have similar predictive power. Moreover, in general, it is beneficial to combine telematics information with classical risk factors because potential interaction may lead to better predictive models. However, the previous statement that *v-a* heatmaps have better predictive power than the 5 classical risk factors (in a GLM) is not confirmed by partitions 2 and 4. For partitions 1 and 4 both the signs of ω and θ^4 switch, leaving the sign of the driving behavior risk factor z_i^4 unchanged. The patterns in the middle column of Fig. 12 are similar to those in Fig. 10, indicating that the hard braking plays a more important role than the hard acceleration. The previous statement of the importance of low speeds are supported by partitions 1 and 2, while partitions 3 and 4 do not violate this statement.

Finally, we perform another data split with a different seed leading to another five training-validation-test partitions. The results for the five partitions are presented in Fig. 13. We conclude with the same findings as those from Fig. 12.

5 Conclusions

We propose two neural networks, a densely connected feed-forward neural network and a convolutional neural network, for extracting driver risk information from telematics car driving data represented by *v-a* heatmaps. The neural networks simultaneously perform feature engineering and regression modeling. Both neural network approaches have a similar predictive performance in our example, however, the convolutional one uses much less parameters. Unlike the black box of the densely connected feed-forward neural network, the convolutional neural network can be interpreted and we explain how the driving behavior risk factor is constructed by the convolutional neural network.

Specific locations in *v-a* heatmaps have their meanings, and we need to make sure that the convolution window has a right size and moves in a sensible way to capture these meanings. Our design of the convolutional neural network targets at detecting similar acceleration patterns in different speed intervals.

As byproducts of our empirical data analysis, we conclude that both the classical risk factors and driving behavior risk factor are needed for claims frequency modeling, and hard braking in low speeds contributes the most to the driving behavior risk factor. Thus, telematics data contains driving style information beyond classical risk factors that is relevant for claim frequency prediction, and on the other hand, classical risk factors as, for instance, regional information may explain certain driving patterns. Letting these two ingredients interact will lead to better predictive models.

Appendix A: `Keras` code

Listing 3: `Keras` code for densely connected feed-forward neural network.

```
build_model_dnn <- function(m1,m2,d1,d2){
  ### input layer
  heatmap <- layer_input(shape = c(6,16,1), dtype = "float32", name = "heatmap")
  vol <- layer_input(shape = c(1), dtype = "float32", name = "vol")

  ### densely connected neural network
  Heatmap_Network = heatmap %>%
    layer_flatten(name = "heatmap_flat") %>%
    layer_dense(units = m1,  activation = "tanh", name = "heatmap_dense1") %>%
    layer_dropout(rate = d1, name = "heatmap_drop1") %>%
    layer_dense(units = m2,  activation = "tanh", name = "heatmap_dense2") %>%
    layer_dropout(rate = d2, name = "heatmap_drop2") %>%
    layer_dense(units = 1, activation = "exponential", name = "heatmap_factor",
        weights = list(array(c(0), dim=c(m2,1)), array(0, dim=c(1))))

  ### response
  Response = list(Heatmap_Network, vol) %>%
    layer_multiply (name = "response", trainable = F)

  ### compile model
  model <- keras_model(inputs = c(heatmap, vol), outputs = c(Response))
  model %>% compile(optimizer = optimizer_adam(), loss = "poisson")
  model
}
```

Listing 4: `Keras` code for convolutional neural network.

```
build_model_cnn <- function(q){
  ### input layer
  heatmap <- layer_input(shape = c(6,16,1), dtype = "float32", name = "heatmap")
  vol <- layer_input(shape = c(1), dtype = "float32", name = "vol")

  ### convolutional neural network
  Heatmap_Network = heatmap %>%
    layer_conv_2d(filters = q, kernel_size = c(6,1), activation = "tanh",
        strides = 1, name = "heatmap_conv1") %>%
    layer_conv_2d(filters = 1, kernel_size = c(1,2), activation = "tanh",
        strides = 2, name = "heatmap_conv2") %>%
    layer_flatten(name = "heatmap_flat") %>%
    layer_dense(units = 1, activation = "exponential", name = "heatmap_factor",
        weights = list(array(c(0), dim=c(8,1)), array(0, dim=c(1))))

  ### response
  Response = list(Heatmap_Network, vol) %>%
    layer_multiply (name = "response", trainable = F)

  ### compile model
  model <- keras_model(inputs = c(heatmap, vol), outputs = c(Response))
  model %>% compile(optimizer = optimizer_adam(), loss = "poisson")
  model
}
```

Appendix B: Approximation of the acceleration pattern $z^3_{i,k}$

Following equations (3.10) and (3.12), and approximating the hyperbolic tangent function by its Taylor's expansion of order 1, $\tanh(x) \approx x$, we have the following approximation:

$$
\begin{aligned}
z^3_{i,k} &= z^2_{i,1,k,1} \\
&= \tanh\left(\theta^2_0 + \sum_{l=1}^{2} \theta^2_{l,1} z^1_{i,1,2k-1,l} + \theta^2_{l,2} z^1_{i,1,2k,l}\right) \\
&\approx \theta^2_0 + \theta^2_{1,1} z^1_{i,1,2k-1,1} + \theta^2_{2,1} z^1_{i,1,2k-1,2} + \theta^2_{1,2} z^1_{i,1,2k,1} + \theta^2_{2,2} z^1_{i,1,2k,2} \\
&= \theta^2_0 + \theta^2_{1,1} \tanh\left(\theta^1_{0,1} + \sum_{j=1}^{6} \theta^1_{j,1} z_{i,j,2k-1,1}\right) + \theta^2_{2,1} \tanh\left(\theta^1_{0,2} + \sum_{j=1}^{6} \theta^1_{j,2} z_{i,j,2k-1,1}\right) + \\
&\quad \theta^2_{1,2} \tanh\left(\theta^1_{0,1} + \sum_{j=1}^{6} \theta^1_{j,1} z_{i,j,2k,1}\right) + \theta^2_{2,2} \tanh\left(\theta^1_{0,2} + \sum_{j=1}^{6} \theta^1_{j,2} z_{i,j,2k,1}\right) \\
&\approx \theta^2_0 + \theta^2_{1,1}\theta^1_{0,1} + \theta^2_{2,1}\theta^1_{0,2} + \theta^2_{1,2}\theta^1_{0,1} + \theta^2_{2,2}\theta^1_{0,2} + \\
&\quad \sum_{j=1}^{6} (\theta^2_{1,1}\theta^1_{j,1} + \theta^2_{2,1}\theta^1_{j,2}) z_{i,j,2k-1,1} + \sum_{j=1}^{6} (\theta^2_{1,2}\theta^1_{j,1} + \theta^2_{2,2}\theta^1_{j,2}) z_{i,j,2k,1} \\
&\approx \theta^2_0 + \theta^2_{1,1}\theta^1_{0,1} + \theta^2_{2,1}\theta^1_{0,2} + \theta^2_{1,2}\theta^1_{0,1} + \theta^2_{2,2}\theta^1_{0,2} + \\
&\quad \sum_{j=1}^{6} (\theta^2_{1,1}\theta^1_{j,1} + \theta^2_{2,1}\theta^1_{j,2} + \theta^2_{1,2}\theta^1_{j,1} + \theta^2_{2,2}\theta^1_{j,2}) \frac{z_{i,j,2k-1,1} + z_{i,j,2k,1}}{2} \\
&= \omega_0 + \sum_{j=1}^{6} \omega_j \frac{z_{i,j,2k-1,1} + z_{i,j,2k,1}}{2},
\end{aligned}
$$

where the last approximation follows from $z_{i,j,2k-1,1} \approx z_{i,j,2k,1} \approx (z_{i,j,2k-1,1} + z_{i,j,2k,1})/2$. This is because the acceleration patterns in the two neighboring speed intervals should be similar.

Appendix C: Figures

See Figs. 11, 12 and 13.

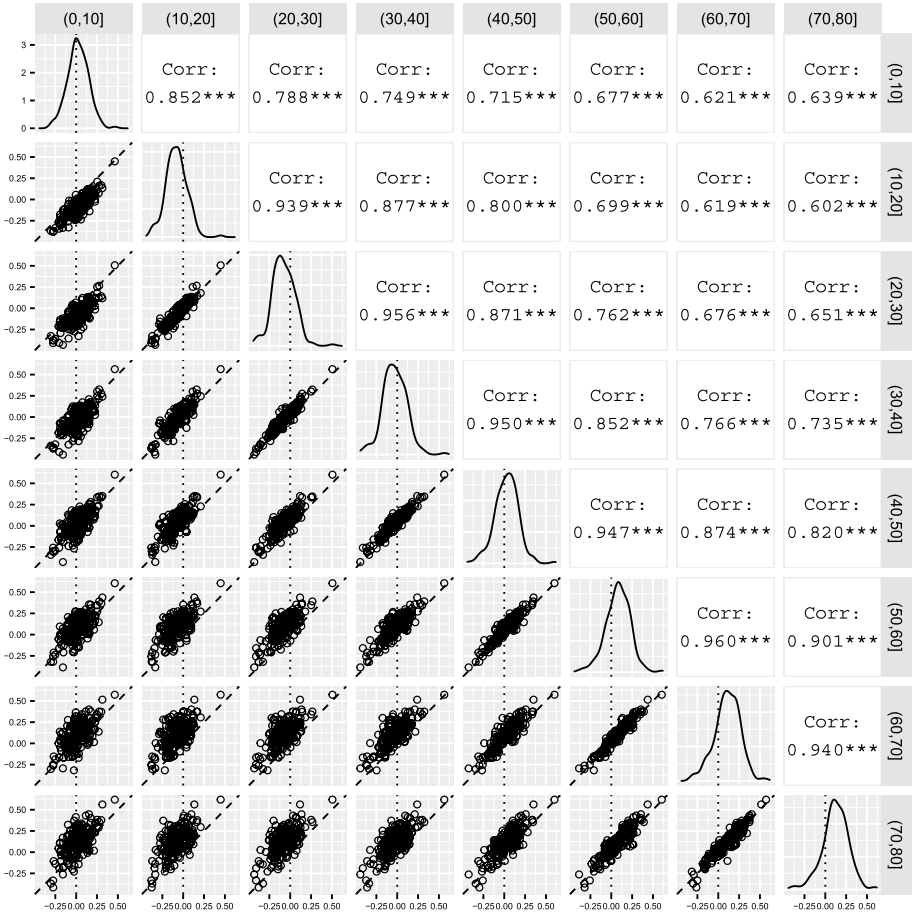

Fig. 11 Pair plots of acceleration patterns $z^3_{i,k}$ in each speed sub-interval $(10(k-1), 10k]$ km/h on the test data $i \in \mathcal{D}_3$

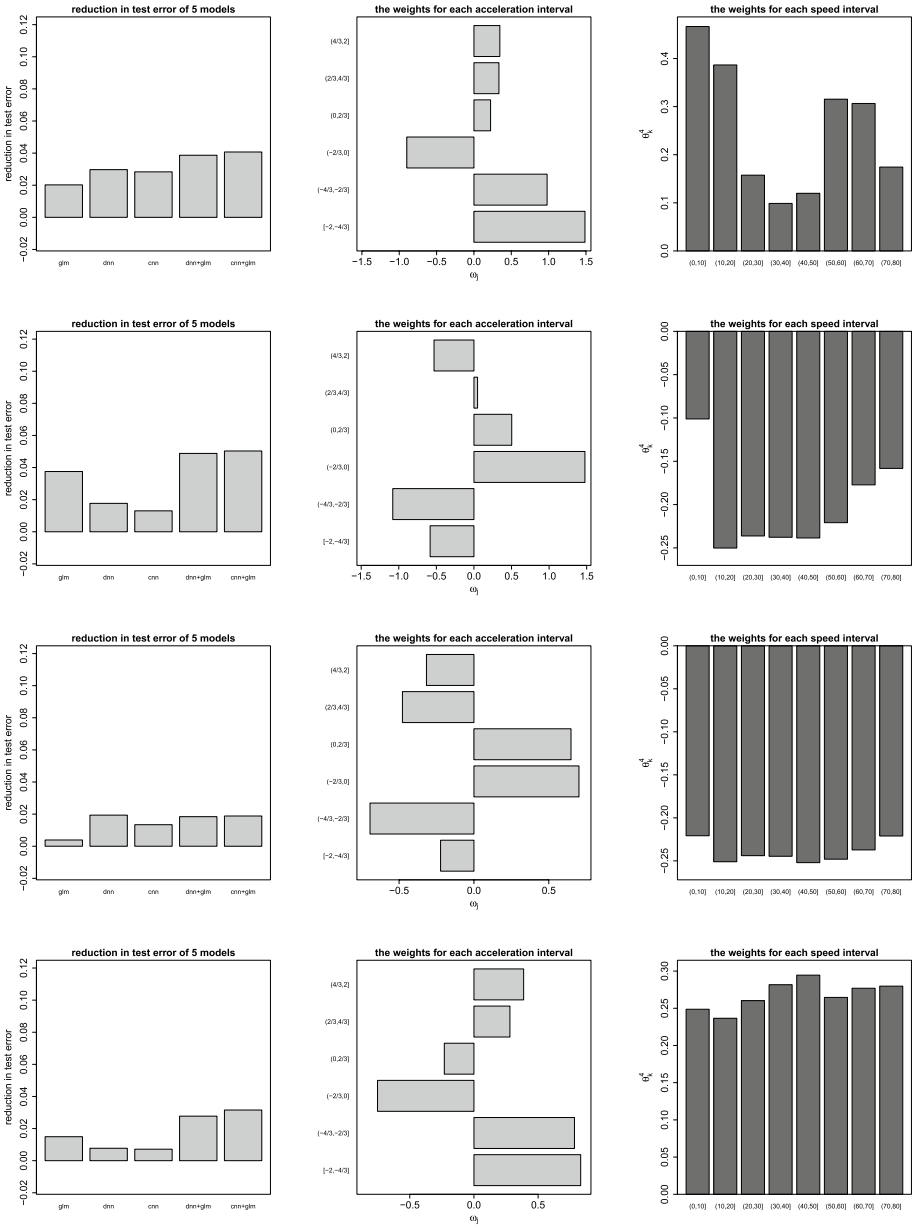

Fig. 12 Each row is for a data partition as shown in Table 5: (left) reduction in test error for glm (2.4), dnn (Listing 1), cnn (Listing 2), dnn + glm (4.1) and cnn + glm (4.2) compared to the homogeneous model (2.6); (middle) weights ω_j for each acceleration sub-interval; (right) weights θ_k^4 for each speed sub-interval

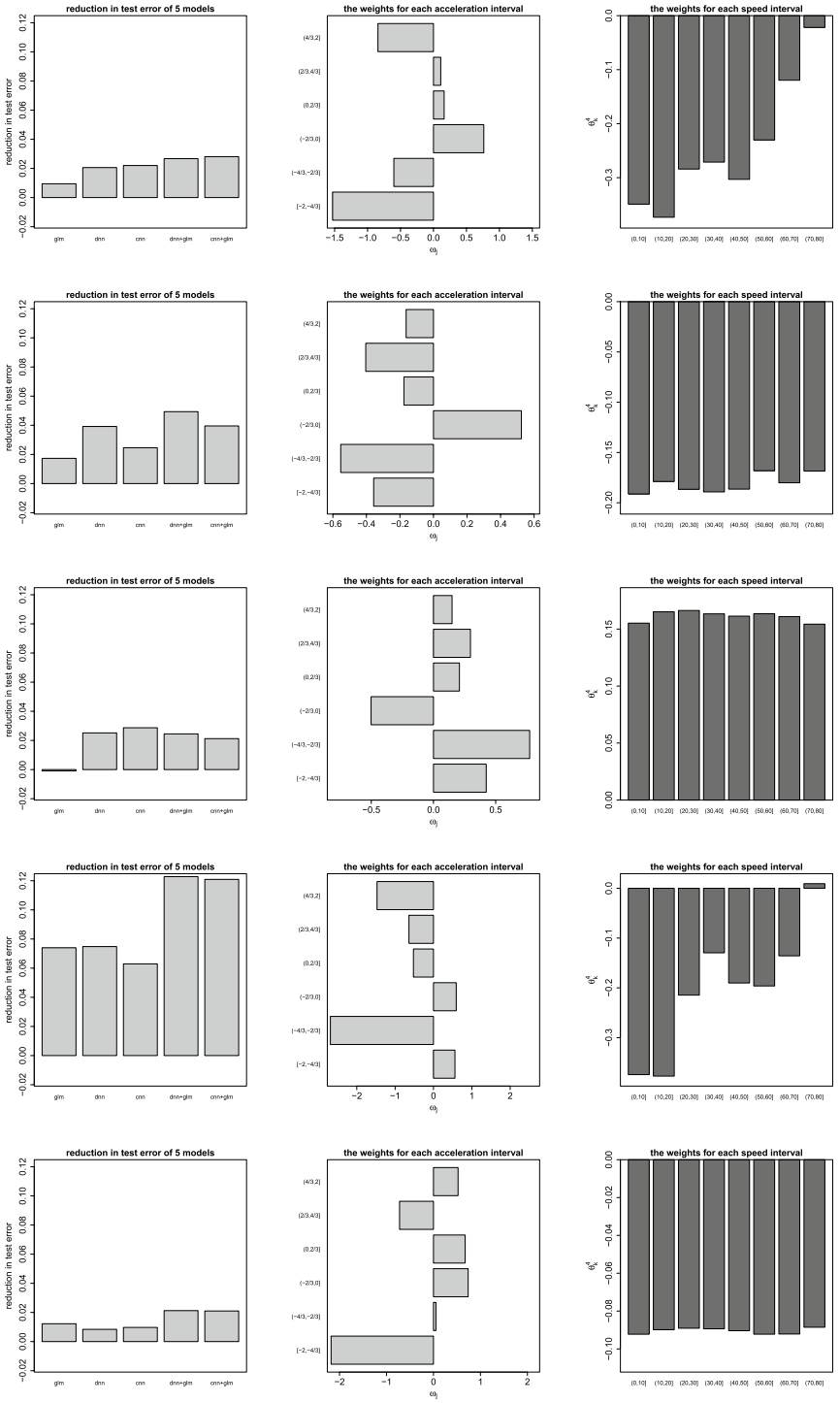

Fig. 13 Another data split using a different seed: (left) reduction in test error for glm (2.4), dnn (Listing 1), cnn (Listing 2), dnn + glm (4.1) and cnn + glm (4.2) compared to the homogeneous model (2.6); (middle) weights ω_j for each acceleration sub-interval; (right) weights θ_k^4 for each speed sub-interval

Acknowledgements Guangyuan Gao gratefully acknowledges financial support from the National Natural Science Foundation of China (71901207).

Funding Open access funding provided by Swiss Federal Institute of Technology Zurich.

References

Ágoston, K. C., & Gyetvai, M. (2020). Joint optimization of transition rules and the premium scale in a bonus-malus system. *ASTIN Bulletin, 50*(3), 743–776.

Ayuso, M., Guillén, M., & Pérez-Marín, A. M. (2016a). Telematics and gender discrimination: Some usage-based evidence on whether men's risk of accidents differs from women's. *Risks* **4/2**, article 10.

Ayuso, M., Guillén, M., & Pérez-Marín, A. M. (2016b). Using GPS data to analyse the distance traveled to the first accident at fault in pay-as-you-drive insurance. *Transportation Research Part C: Emerging Technologies, 68,* 160–167.

Ayuso, M., Guillén, M., & Nielsen, J. P. (2019). Improving automobile insurance ratemaking using telematics: Incorporating mileage and driver behaviour data. *Transportation, 46,* 735–752.

Boucher, J.-P., Côté, S., & Guillén, M. (2017). Exposure as duration and distance in telematics motor insurance using generalized additive models. *Risks* **5/4**, article 54.

Boucher, J.-P., & Inoussa, R. (2014). A posteriori ratemaking with panel data. *ASTIN Bulletin, 44*(3), 587–612.

Boucher, J.-P., & Pigeon, M. (2018). A claim score for dynamic claim counts modeling. arXiv https://arxiv.org/abs/1812.06157.

Brouhns, N., Guillén, M., Denuit, M., & Pinquet, J. (2003). Bonus-malus scales in segmented tariffs with stochastic migration between segments. *The Journal of Risk and Insurance, 70*(4), 577–599.

Chollet, F., & Allaire, J. J. (2018). *Deep Learning with R*. Manning Publication.

De Pril, N. (1978). The efficiency of a bonus-malus system. *ASTIN Bulletin, 10*(1), 59–72.

Denuit, M., Guillén, M., & Trufin, J. (2019). Multivariate credibility modelling for usage-based motor insurance pricing with behavioural data. *Annals of Actuarial Science, 13*(2), 378–399.

Denuit, M., Maréchal, X., Pitrebois, S., & Walhin, J.-F. (2007). *Actuarial Modelling of Claim Counts: Risk Classification. Credibility and Bonus-Malus Systems*. Wiley.

Ferrario, A., Noll, A., & Wüthrich, M. V. (2018). Insights from inside neural networks. *SSRN*, Abstract Id: 3226852.

Gao, G., Meng, S., & Wüthrich, M. V. (2019). Claims frequency modeling using telematics car driving data. *Scandinavian Actuarial Journal, 2019*(2), 143–162.

Gao, G., & Wüthrich, M. V. (2019). Convolutional neural network classification of telematics car driving data. *Risks* **7/1**, article 6.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Henckaerts, R., Antonio, K., Clijsters, M., & Verbelen, R. (2018). A data driven binning strategy for the construction of insurance tariff classes. *Scandinavian Actuarial Journal, 2018*(8), 681–705.

Henckaerts, R., Côté, M.-P., Antonio, K., & Verbelen, R. (2019). Boosting insights in insurance tariff plans with tree-based machine learning. arXiv, 1904.10890.

Ho, S.-H., Wong, Y.-D., & Chang, V.W.-C. (2014). Developing Singapore driving cycle for passenger cars to estimate fuel consumption and vehicular emissions. *Atmospheric Environment, 97,* 353–362.

Huang, Y., & Meng, S. (2019). Automobile insurance classification ratemaking based on telematics driving data. *Decision Support Systems* **127**, article 113156.

Hung, W. T., Tong, H. Y., Lee, C. P., Ha, K., & Pao, L. Y. (2007). Development of practical driving cycle construction methodology: a case study in Hong Kong. *Transportation Research Part D: Transport and Environment, 12*(2), 115–128.

Kamble, S. H., Mathew, T. V., & Sharma, G. K. (2009). Development of real-world driving cycle: case study of Pune, India. *Transportation Research Part D: Transport and Environment, 14*(2), 132–140.
Lemaire, J. (1995). *Bonus-Malus Systems in Automobile Insurance*. Kluwer Academic Publisher.
Lemaire, J., Park, S. C., & Wang, K. (2016). The use of annual mileage as a rating variable. *ASTIN Bulletin, 46*(1), 39–69.
Lee, S. C. K. (2021). Addressing imbalanced insurance data through zero-inflated Poisson regression with boosting. *ASTIN Bulletin, 51*(1), 27–55.
Loimaranta, K. (1972). Some asymptotic properties of bonus systems. *ASTIN Bulletin, 6*(3), 233–245.
Paefgen, J., Staake, T., & Fleisch, E. (2014). Multivariate exposure modeling of accident risk: insights from pay-as-you-drive insurance data. *Transportation Research Part A: Policy and Practice, 61,* 27–40.
Richman, R. (2020a). AI in actuarial science – a review of recent advances – part 1. *Annals of Actuarial Science*. https://doi.org/10.1017/S1748499520000238.
Richman, R. (2020b). AI in actuarial science – a review of recent advances – part 2. *Annals of Actuarial Science*. https://doi.org/10.1017/S174849952000024X.
Sun, S., Bi, J., Guillén, M., & Pérez-Marín, A.M. (2020). Assessing driving risk using internet of vehicles data: an analysis based on generalized linear models. *Sensors* **20/9**, article 2712.
Verbelen, R., Antonio, K., & Claeskens, G. (2018). Unraveling the predictive power of telematics data in car insurance pricing. *Journal of the Royal Statistical Society: Series C (Applied Statistics), 67,* 1275–1304.
Verschuren, R. M. (2021). Predictive claim scores for dynamic multi-product risk classification in insurance. *ASTIN Bulletin, 51*(1), 1–25.
Weidner, W., Transchel, F. W. G., & Weidner, R. (2016). Classification of scale-sensitive telematic observables for riskindividual pricing. *European Actuarial Journal, 6*(1), 3–24.
Weidner, W., Transchel, F. W. G., & Weidner, R. (2017). Telematic driving profile classification in car insurance pricing. *Annals of Actuarial Science, 11*(2), 213–236.
Wiatowski, T., & Bölcskei, H. (2018). A mathematical theory of deep convolutional neural networks for feature extraction. *IEEE Transactions on Information Theory, 64*(3), 1845–1866.
Wüthrich, M. V., & Merz, M. (2019). Editorial: Yes, we CANN! *ASTIN Bulletin, 49*(1), 1–3.
Yang, Y., Qian, W., & Zou, H. (2018). Insurance premium prediction via gradient tree-boosted Tweedie compound Poisson models. *Journal of Business and Economic Statistics, 36*(3), 456–470.
Zhang, W., Itoh, K., Tanida, J., & Ichioka, Y. (1990). Parallel distributed processing model with local space-invariant interconnections and its optical architecture. *Applied Optics, 29*(32), 4790–4797.
Zhang, W., Tanida, J., Itoh, K., & Ichioka, Y. (1988). Shift invariant pattern recognition neural network and its optical architecture. In: Proceedings of the Annual Conference of the Japan Society of Applied Physics, 6p-M-14, 734.