Quantifying the Uncertainty of Mobility Flow Predictions Using Gaussian Processes

Aike Steentoft[1], Bu-Sung Lee[1] and Markus Schläpfer[1,2,3*]

[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore.
[2]Centre for Regional Economic Development, University of Bern, Bern, Switzerland.
[3]Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, USA.

*Corresponding author. E-mail: m.schlaepfer@columbia.edu

Abstract

The ability to understand and predict the flows of people in cities is crucial for the planning of transportation systems and other urban infrastructures. Deep-learning approaches are powerful since they can capture non-linear relations between geographic features and the resulting mobility flow from a given origin location to a destination location. However, existing methods are not able to quantify the uncertainty of the predictions, which limits their interpretability and thus their use for practical applications in urban infrastructure planning. To that end, we propose a Bayesian deep-learning approach that formulates deep neural networks as Gaussian processes and integrates automatic variable selection. Our method provides uncertainty estimates for the predicted origin-destination flows while also allowing to identify the most important geographic features that drive the mobility patterns. The developed machine learning approach is applied to large-scale taxi trip data from New York City.

Keywords: Mobility, Bayesian deep learning, Smart cities, Transportation system planning

Version: This version of the article has been accepted for publication after peer review but is not the Version of Record and does not reflect post-acceptance improvements. The Version of Record is available online at: https://doi.org/10.1007/s11116-023-10406-z. Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use.

1 Introduction

Transportation systems and other urban infrastructures need to be continuously adapted to cope with an increasing number of grand challenges such as population growth, climate change, disruptive technological innovations and, more recently, the global pandemic [1, 2]. At the same time, infrastructure developments typically affect many people, cost lots of resources and are difficult to reverse [3]. As a consequence, planners and engineers, responsible for the sustainable design of these systems, need to base their decisions on a profound, science-based understanding of how people make use of urban space and how their daily mobility patterns are affected by local characteristics [4–6].

To that end, mathematical models help to quantify the relation between urban features and the mobility flows between different origin and destination locations [7]. Of particular importance are gravity models [8–10] that predict the number of people traveling from an origin i to a destination j as $T_{ij} = c\, m_i m_j f(r_{ij})$, where c is a constant, m_i and m_j are key local attributes, and $f(\cdot)$ is a decreasing function of the distance r_{ij} between the two locations. A common choice is $m_i = P_i^{\alpha}$ and $m_j = P_j^{\beta}$, where P_i and P_j are the population sizes of the locations i and j, and α and β are parameters that are calibrated with data. Gravity models can be extended to include additional variables such as travel time [11]. Typically, the function $f(\cdot)$ takes the form of an exponential or power-law $f(r_{ij}) = r_{ij}^{-\gamma}$ where γ is an additional parameter that needs to be calibrated. While the functional form of the gravity models is simple and explainable, it may also lead to high prediction errors since it is not able to reflect many interacting factors such as varying trip purposes over the course of a day or socio-economic characteristics of the travelers [12]. Being a more fundamental limitation, gravity models so far do not provide uncertainty estimates for their predictions. In other terms, decision makers do not know when the model is uncertain about its predictions and whether it might even be guessing at random, which is highly undesirable for infrastructure planning applications. The same limitations also apply to intervening opportunities models which, besides gravity models, constitute the second important class of traditional mobility flow models [13, 14].

Several machine learning approaches have been developed to address some of these limitations. Examples are decision tree models [15, 16] and feed-forward neural networks [15–19] that are able to outperform the traditional mobility flow models in terms of prediction performance. More recently, various deep learning techniques have been applied to the prediction of origin-destination flows from location-specific urban features (also called 'flow generation') [20]. Particularly relevant is Deep Gravity [21], which is a deep neural network model for predicting origin-destination flows based on an extensive set of geographic features such as land use, road network, food and health facilities. Explainability is added through using SHAP (SHapley Additive exPlanation), allowing to understand how strongly the input features contribute to the mobility predictions. The strength of such approaches lies in pattern recognition, where the non-linearity of the models allows to improve

the prediction performance through detecting complex regularities in the data. However, similar to traditional models, they do not provide any uncertainty estimates associated with individual predictions.

To address this gap, we introduce a deep learning approach that i) is able to consider many input variables with complex relational structure, ii) is interpretable with respect to the relevance of the individual input variables and iii) is interpretable with respect to the uncertainty of its predictions. The approach uses fully Bayesian inference on deep neural networks that are formulated as Gaussian processes.

2 Methodology

2.1 Bayesian deep learning using Gaussian process priors

Deep neural networks have a high prediction performance in pattern recognition problems [22] since they are able to capture complex, non-linear dependencies. However, after training a deep neural network, it is difficult to obtain uncertainty estimates of its predictions [23]. This is because deep learning models are typically deterministic functions where parameter values and predictions are point estimates rather than probability distributions.

Bayesian deep learning is the principled approach of dealing with these uncertainties by defining deep learning models in a probabilistic way [24, 25]. The resulting models are typically more robust against overfitting, provide uncertainty estimates for parameters and predictions and can handle small data sizes, while being as straightforward to train as traditional deep learning models [26]. Such models are therefore of great value for critical applications where uncertainty information is required. While the intersection between deep learning and Bayesian models has recently received great attention in the field of probabilistic mobility modelling (e.g., [27–30]), its application to the generation of origin-destination flows on the basis of urban features has remained underexplored.

Bayesian deep learning requires first, the definition of a probabilistic model and second, the inference of the model parameters from the observed data. In line with the Bayesian paradigm, we have to define a prior distribution as well as a likelihood function. The prior distribution over the network parameters θ gives a prior distribution over parametric functions $f(x;\ \theta)$, where $f(\cdot)$ is the network function and x is the network input. The weights and biases in each layer are often given standard Gaussian prior distributions. The likelihood function is treated as in Generalized Linear Models where the neural network is used instead of a linear model. The main difficulty in Bayesian deep learning is the inference part due to a large amount of parameters which leads to very high-dimensional, intractable posterior probability distributions [26]. This problem has been tackled through stochastic inference approaches [31] such as Markov chain Monte Carlo (MCMC) methods. These methods sample from the true posterior distribution and are characterized by the theoretical convergence to the true model. But in the context of neural network models

MCMC methods are able to explore the posterior only in a small neighborhood that is sensitive to the initial conditions. Sampling then may produce useful predictive models but it still fails to sample correctly from the whole posterior. Alternatively, deterministic approaches [32, 33] approximate the intractable posterior distribution using some tractable form but do not quantify how close the posterior approximation is to the true model [34].

Recently, the equivalence between infinitely wide deep neural networks (which are deterministic functions) and Gaussian processes (which are probability distributions over functions) has been derived [35, 36]. This approach promises to overcome many of the above challenges by providing exact and efficient Bayesian inference. This is an alternative to deterministic or stochastic inference with the original neural network model where the prior distribution of the neural network is reformulated as a Gaussian process prior. Concretely, given a feed-forward neural network and placing Gaussian distributions over the weights, then in the limit of infinite network width the model becomes a Gaussian process with the kernel being dependent on the network architecture and prior specification. This reformulation then allows for Bayesian prediction with deep neural networks where the computation only requires building the kernel and straightforward matrix computations without the need for stochastic gradient-based training. Indeed, Lee et al. [35] used Gaussian processes for Bayesian inference on regression tasks and compared their performance to stochastic gradient training. They found that the neural network accuracy approaches that of the Gaussian process with increasing layer width, that the Gaussian process predictions typically outperform the neural networks with finite width, and that the Gaussian process uncertainty is strongly correlated with the prediction error. This probabilistic formulation of an infinitely wide deep neural network as a Gaussian process has been termed Neural Network Gaussian Process (NNGP) [35]. A general limitation is the computational complexity which scales cubically in the number of data points due to the matrix inverse problem in Gaussian process inference [37].

Thus, for the neural network model, we consider a feed-forward fully-connected architecture with L hidden layers and N_l hidden units in layers $l = 1, \ldots, L$. The input for the model is $x \in \mathbb{R}^D$ and the output is $z \in \mathbb{R}$. In each layer l and unit i, x_i^l is the value after the activation and z_i^l is the value after the linear transformation and before the next activation. In the case of the input, x_i^α is the value of variable i for a particular input α. The non-linear transformation is denoted by $\phi(\cdot)$. The weights W_{ij}^l within a layer have the following hierarchical prior

$$W_{ij}^l \sim \mathcal{N}(0, \frac{\sigma_w^2}{N_l})$$
$$\sigma_w^{-2} \sim \text{Gamma}(\alpha_w, \beta_w),$$

and the biases b_i^l within a layer have the following hierarchical prior

$$b_i^l \sim \mathcal{N}(0, \sigma_b^2)$$
$$\sigma_b^{-2} \sim \text{Gamma}(\alpha_b, \beta_b)$$

The correspondence between deep neural networks and Gaussian processes can be seen in the network function [35]

$$z_i^l(x) = b_i^l + \sum_{j=1}^{N_l} W_{ij}^l x_j^l(x), \quad x_j^l(x) = \phi\left(z_j^{l-1}(x)\right).$$

By induction it can be shown that if N_{l-1} goes to infinity then the input to layer l is a Gaussian process and therefore $z_i^l(x)$ is a Gaussian process with

$$\begin{aligned} z_i^l &\sim \text{GP}(\mu^l, k^l) \\ \mu^l &= \text{E}\left[z_i^l(x)\right] = 0 \\ k^l(x, x_*) &= \text{cov}\left[z_i^l(x), z_i^l(x_*)\right] = \text{E}\left[z_i^l(x) z_i^l(x_*)\right] \\ &= \sigma_b^2 + \sigma_w^2 \text{E}_{z_j^{l-1} \sim \text{GP}(0, k^{l-1})}\left[\phi(z_j^{l-1}(x))\phi(z_j^{l-1}(x_*))\right] \\ &= \sigma_b^2 + \sigma_w^2 F\left(k^{l-1}(x, x_*), k^{l-1}(x, x), k^{l-1}(x_*, x_*)\right) \end{aligned}$$

where F is a deterministic function that describes the relationship between k^l and k^{l-1}. k^l can then be solved analytically, e.g. if $\phi(\cdot)$ is the Rectified Linear Unit (ReLU) non-linearity then k^l becomes the arccosine kernel [38] and k^L can be computed iteratively.

2.2 Sparse learning for variable selection

To improve the interpretability of the inputs to the model, we additionally use sparse learning. In the typical sparse learning setting, the model allows for many covariates that are potentially relevant for prediction. Sparsity then has the underlying assumption that only a subset of these covariates is relevant but we don't know beforehand which one. By selecting only the relevant variables, sparse learning can help to better understand the process that generated the data, it can reduce measurement requirements, and it can improve prediction performance especially if the number of data points is small compared to the number of covariates. To enable Bayesian sparsity, we need a prior that collapses all the posterior mass to either relevance or irrelevance. Here, the regularized horseshoe prior [39], which is an extension of the original horseshoe prior [40], is particularly useful since the performance of the model is not very sensitive to the choices in the prior parameters (in contrast to other approaches such as Automatic Relevance Determination) [25].

To our knowledge, no research has been done on integrating automatic variable selection in the formulation of deep neural networks as Gaussian processes. To that end, we integrate the horseshoe prior into the neural network

kernel of the Gaussian process by introducing additional hyperparameters to the kernel. Specifically, we propose a horseshoe prior on the weight parameters connected to the input variables which induces sparsity in the first layer ($l = 0$) of the corresponding neural network [39],

$$\begin{aligned} W_{ij}^0 \mid \lambda_j, \tau, \sigma_0 \sim \mathcal{N}(0, \tau^2 \tilde{\lambda}_j^2), \quad \tilde{\lambda}_j^2 = \frac{\sigma_0^2 \lambda_j^2}{\sigma_0^2 + \tau^2 \lambda_j^2}, \\ \lambda_j \sim \mathcal{C}^+(0, 1), \\ \sigma_0^{-2} \sim \text{Gamma}(\alpha_0, \beta_0), \quad \alpha_0 = \nu_0/2, \quad \beta_0 = \nu_0 s^2/2 \\ \tau \sim \mathcal{C}^+(0, \sigma_g^2), \end{aligned}$$

where all weights W_{ij}^0 that connect to the same input component x_j are drawn from a scale mixture of normal distributions with variance $\tau^2 \tilde{\lambda}_j^2$, where τ is the global shrinkage parameter that shrinks all parameters to zero while $\tilde{\lambda}_j$ is the local shrinkage parameter that allows individual parameters to escape this shrinkage if it is useful for the prediction. The parameters τ and λ_j are themselves unknown. The prior for τ ensures that the data can inform the global shrinkage beyond the initial guess σ_g. The heavy-tailed Cauchy prior of λ_j allows individual parameters to go to large values and go above the global shrinkage τ if needed. By using $\tilde{\lambda}_j^2$ instead of λ_j^2 directly, the main problem of the original horseshoe prior can be solved which is that parameters that escape the shrinkage are almost unregularized and therefore can go to very large values. After marginalizing, the Gamma distribution on σ_0^{-2} then results in a Student-$t_{\nu_0}(0, s_0^2)$ for the parameters far from zero. After training, some of the hyperparameters associated to an input variable will be much smaller than others, indicating that the corresponding input variable is not relevant for the predictions.

Next, we convert the above formulation of the prior distribution for the first layer of the neural network into the formulation of the Gaussian process prior for deep neural networks. With $W_{ij}^0 \sim \mathcal{N}(0, \tau^2 \tilde{\lambda}_j^2)$ and $b_i \sim \mathcal{N}(0, \sigma_b^2)$ the base case k^0 becomes

$$\begin{aligned} k^0(x, x_*) &= \text{cov}\left[z_i^0(x), z_i^0(x_*)\right] = \text{E}\left[z_i^0(x) z_i^0(x_*)\right] \\ &= \sigma_b^2 + \tau^2 \sum_{j=1}^{D} \tilde{\lambda}_j^2 x_j x_{*j} \end{aligned}$$

We can now utilize the recursion relating k^1 and k^0 to complete the iterative series of computations to obtain k^L for the Gaussian process that describes the network's final output.

Consider a dataset $\{(x^1, y^1), \ldots, (x^N, y^N)\}$ with N input-target pairs, for example in mobility flow modeling, N origin-destination pairs where $\boldsymbol{x}$ are the input variables such as number of people living and working at origin and destination and $\boldsymbol{y}$ are the flows between the origins and destinations. The goal

is to construct a distribution over functions $z(x)$ that can be used to make predictions at a test point x_*. The distribution over functions is modeled with the above described Gaussian process model and for the noise model we assume a Gaussian centered at $\boldsymbol{z}$ which allows for exact Bayesian inference with pure matrix calculation.

The deep neural network kernel and the noise model are fully determined by the set of parameters $\theta = \{\sigma_w, \sigma_b, \tau, \sigma_0, \lambda, \sigma_n\}$. Other parameters such as the depth of the network are not modelled explicitly and we simply test for different configurations. The log of the posterior distribution over the parameter space is then $\log p(\theta \mid \boldsymbol{x}, \boldsymbol{y}) \propto \log p(\theta) + \log p(\boldsymbol{y} \mid \theta, \boldsymbol{x})$. The prior of the noise variance is $\sigma^{-2} \sim \text{Gamma}(\alpha_n, \beta_n)$ and the calculation of the log marginal likelihood is given in [41].

The entire computations including the construction of the kernel are differentiable. This makes it possible to use the Hamiltonian Monte Carlo (HMC) algorithm [42] for finding the best hyperparameter configuration of the Gaussian process kernel. The output is a set of M samples $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_M\}$ that can be used to compute expectation values of any function of interest. For Bayesian prediction we can draw for each θ_i samples from the Gaussian process model $z_{*ij} \sim p(z_* \mid \theta_i, x_*)$ where $i = 1, \ldots, M$ are the MCMC samples of θ and $j = 1, \ldots, L$ are the samples of z_* for each θ_i. The samples z_{*ij} can then be used to compute summaries such as mean and standard deviation.

Taken together, we refer to our proposed framework as *Horseshoe Neural Network Gaussian Process* (H-NNGP). Its design allows for fully Bayesian inference of deep neural networks. With the formulation as a Gaussian process we can explicitly model the uncertainty in the dataset, as well as in the prediction function which is important for critical decision making in the urban context. Additionally, the introduced hyperparameters in the covariance function of the Gaussian process allow a principled way of automatic variable selection. Finally, the framework can replicate the linear gravity model (simply by setting the number of hidden layers to zero), as well as powerful non-linear models (with increasing number of hidden layers), and there is a theoretical guarantee that the posterior parameters (and thereby the variable importance) are correct.

2.3 Assessment of H-NNGP on synthetic data

To assess the H-NNGP model in a controlled setting, we test it on synthetic data. To that end, we created a synthetic dataset with the 'Madelon' template that was proposed in the NIPS 2003 variables selection challenge [43]. By adjusting the parameters of the template it is possible to generate versions of the dataset that allow to focus on specific variable selection challenges. Madelon is a two-class classification dataset with continuous input variables and a binary target variable. Following [43], we generate one small problem ('s-Madelon'), and one large problem ('l-Madelon'), see table 1. The l-Madelon is substantially more difficult than s-Madelon due to the larger number of variables and a smaller ratio of samples to informative variables. Moreover,

Table 1 Parameter choices for the synthetic datasets

Parameter	s-Madelon	l-Madelon
Number of classes	2	2
Number of clusters per class	2	16
Number of samples per cluster	50	25
Number of informative variables	2	5
Number of redundant variables	2	5
Number of repeated variables	2	10
Number of noninformative variables	14	80
Factor multiplying hypercube size	2	2
Fraction of random labels	0.01	0.01

the s-Madelon is an XOR ('exclusive or') classification problem [43], which can not be solved by linear models [44], while the l-Madelon does not have a clear XOR structure. Together, the datasets cover different challenges related to the number of redundant, repeated, and non-informative variables, the number of training samples, and the nonlinearity of the data.

2.3.1 Model configurations

In the experiments the classification task is formulated as a regression task [45] because this enables the use of a Gaussian noise model and therefore allows exact inference for the H-NNGP model. While less principled, least-squares classification has been shown to perform well [46]. Similar to [35], the class labels are encoded as one-hot zero mean multidimensional regression targets. We draw 100 training samples and 100 test samples for the s-Madelon experiment, and 400 training samples and 400 test samples for the l-Madelon experiment. The training sets are used for the inference part and the test sets are used to report the performance.

The H-NNGP performance is tested for different choices of (L, σ_g^2). The hyperparameter network depth, L, was manually set and tested for $\{0, 1, 3\}$, where $L = 0$ has zero hidden layer and is equivalent to a Gaussian Process with a linear kernel, $L = 1$ represents a basic non-linear model, and $L = 3$ represents a more complex non-linear model. The hyperparameter σ_g^2 for the prior of the global shrinkage parameter τ was tested on log scale for $\{0.01, 1, 100\}$. The hyperparameters for the remaining priors are fixed: (i) $\alpha_w = 1$ and $\beta_w = 10^{-2}$ which gives a weakly informative prior for σ_w^{-2}, (ii) $\alpha_b = 1$ and $\beta_b = 10^{-2}$ which gives a weakly informative prior for σ_b^{-2}, (iii) $\nu_0 = 1$ and $s_0 = 1$ which results in a weakly informative Student-$t_{\nu_0}(0, s_0^2)$ prior for the parameters far from zero, and (iv) $\alpha_n = 10^{-6}$ and $\beta_n = 10^{-6}$ which gives an uninformative prior for the noise σ_n^{-2}.

The HMC algorithm is used to find the best hyperparameter configuration of the Gaussian process kernel, in particular the variance of the global shrinkage parameter. In each experiment the Markov chain is started at the maximum a posteriori which is found with the Adam optimizer [47].

2.3.2 Results

The first part of results concerns the prediction performance in terms of the accuracy of the binary classification task, see table 2. For s-Madelon the linear models with zero hidden layers have an accuracy close to 50 percent which, for a binary classification task, corresponds to guessing at random and is not surprising given the XOR classification task [44]. The non-linear models reach almost perfect accuracy considering the noise in the data. The picture is similar for l-Madelon. The linear models perform slightly better than guessing at random since it's not a perfect XOR task anymore. The non-linear models perform slightly worse than for the s-Madelon due to the higher complexity of the classification task. Noteworthy, the classification accuracy is insensitive to the number of hidden layers as long as the model is non-linear ($L \geq 1$) and the classification accuracy is also insensitive to different global shrinkage parameters. This is likely due to the simplicity of the synthetic datasets where non-linear patterns can be detected relatively easily. Interestingly, the hyperparameter σ_g^2 that shapes the prior of the global shrinkage parameter τ has no significant effect on the prediction accuracy. This suggests that the parameter τ is mainly informed by the data and less effected by the prior.

The second part of results concerns the uncertainty estimation performance. An advantage of the Bayesian formulation is that it allows to assign a variance to each test point prediction which encodes how certain the model is about a prediction. Figure 1 shows the correlation between uncertainty estimates and prediction error. Test points with a higher uncertainty estimate tend to have higher prediction error. This shows that the model seems to 'know when it doesn't know'.

The third part of results concerns the variable selection performance. Variables are selected based on the magnitude of the local shrinkage parameters λ, where a large λ_i indicates that the connected variable i is relevant for explaining the observed data. The magnitude does not represent the relative relevance of individual variables but can only be used to distinguish between

Table 2 Test accuracy for various model configurations for the Madelon datasets. The Gaussian process models are specified by L and σ_g^2 (e.g., H-NNGP-0-1 corresponds to $L = 0$ and $\sigma_g^2 = 1$).

Model	s-Madelon	l-Madelon
H-NNGP-0-1	0.47	0.58
H-NNGP-0-100	0.47	0.58
H-NNGP-1-0.01	0.96	**0.87**
H-NNGP-1-1	**0.98**	**0.87**
H-NNGP-1-100	**0.98**	**0.87**
H-NNGP-3-0.01	0.94	**0.87**
H-NNGP-3-1	0.95	**0.87**
H-NNGP-3-100	0.94	**0.87**

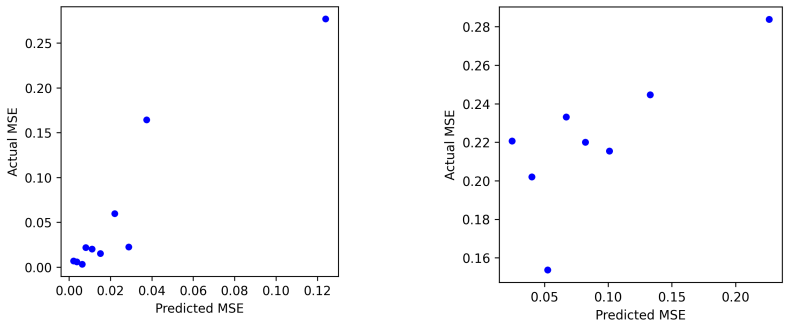

Fig. 1 Test points are sorted according to the predicted mean squared error (MSE) and then binned (x-axis). The predicted MSE is equivalent to the mean of the variance of the Gaussian noise model after inference. The y-axis is the actual MSE for the test points in that bin. Left: small Madelon dataset with bin size 10. Right: large Madelon dataset with bin size 50.

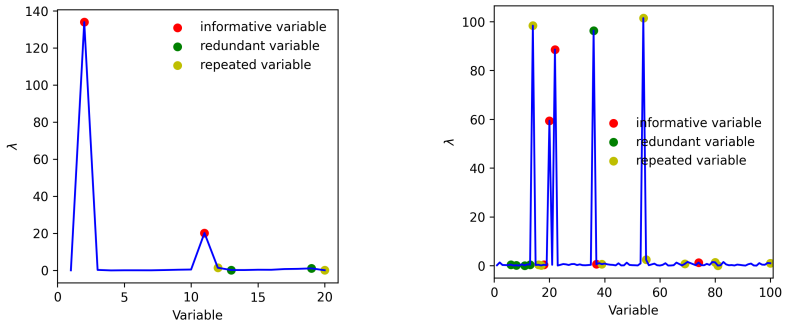

Fig. 2 The blue line shows the posterior mean value of the local shrinkage parameters λ that determine the shrinkage of the connected variables, the colored dots show which of the variables are true informative, redundant, and repeated, (left) for a selected model for the small Madelon dataset, (right) for a selected model for the large Madelon dataset.

relevant and irrelevant. Figure 2 shows the posterior values for all local shrinkage parameters. The model correctly selects all variables that contain helpful information, either by being informative (directly or by repeating an informative variable that was not selected) or by a combination of informative variables (redundant), without selecting non-informative variables.

It should be added that the high computational complexity of Gaussian processes (section 2.1) is reflected in relatively long CPU times. For instance, using a 2.4 GHz 8-Core Intel Core i9, the CPU times for the inference and prediction for l-Madelon with $L = 3$ and $\sigma_g^2 = 1$ were 1257 s and 176 s, respectively. However, relatively long CPU times are not a priority issue for the planning of transportation systems and other urban infrastructures, since these applications typically do not require predictions close to real time [48].

3 Prediction of origin-destination flows

3.1 Data

We apply the developed machine learning approach, H-NNGP, to the prediction of intra-city origin-destination (OD) flows. The input variables are locational attributes (urban features) of the origin and destination, as well as of the locations in between, while the target variable is the flow of individuals between the origin and the destination. We test the model on a GPS-based taxi trip dataset from New York City where the task is to predict number of trips between census tracts based on a set of census tract variables (e.g., population size, number of jobs) and a set of 'link' variables (e.g., geographic distance, number of intervening jobs). We opt for a quasi experimental setting where for the training set we randomly select mobility flows from the year 2014 and for the test set we randomly select mobility flows from the year 2015. We additionally analyze how the flow dynamics change over the course of a day. The H-NNGP model is evaluated according to prediction performance, uncertainty estimation, and variable selection.

For the experiments, we create a dataset that contains i) aggregated OD flows and ii) attributes data that describe the census tracts. The workflow to build the dataset is as follows:

1. The city is divided into 2,167 census tracts [49]; each census tract represents an origin and destination in the OD-matrix.
2. The pick-up and drop-off times and locations are collected for all yellow and green taxis for the years 2014 and 2015 [50].
3. The latitude and longitude of the pick-up and drop-off locations are converted into census tract IDs.
4. The trips are aggregated spatially (same origin and destination) and temporally (same year, weekday chunk, and hour chunk). Here weekday chunk subdivides into weekdays/weekends and hour chunk subdivides into 2-hour slots starting at 11pm (starting at an odd number subdivides the day into more 'natural' 2-hour slots). Figure 3, top panel, shows the mapping of a selection of binned OD flows. The frequency of trips for an OD pair was highest in Manhattan and decreases with increasing distance from the city center; a noteworthy exception are the airports outside the city center. Figure 3, bottom panel, shows the probability distribution of OD flows aggregated over all temporal cases. The low-frequency flows are heavily over-represented in the dataset and there is only a relatively small share of high-frequency flows.
5. The location attributes are selected with a heuristic approach. First, various trip purposes are defined (e.g. home-work); second, for each trip purpose, factors are included that relate to i) trip production, e.g. number of jobs with a specific income range located at the origin, ii) trip attraction, e.g. number of jobs with a specific income range located at the destination and iii) interaction affinity, e.g. distance. The list of attributes is

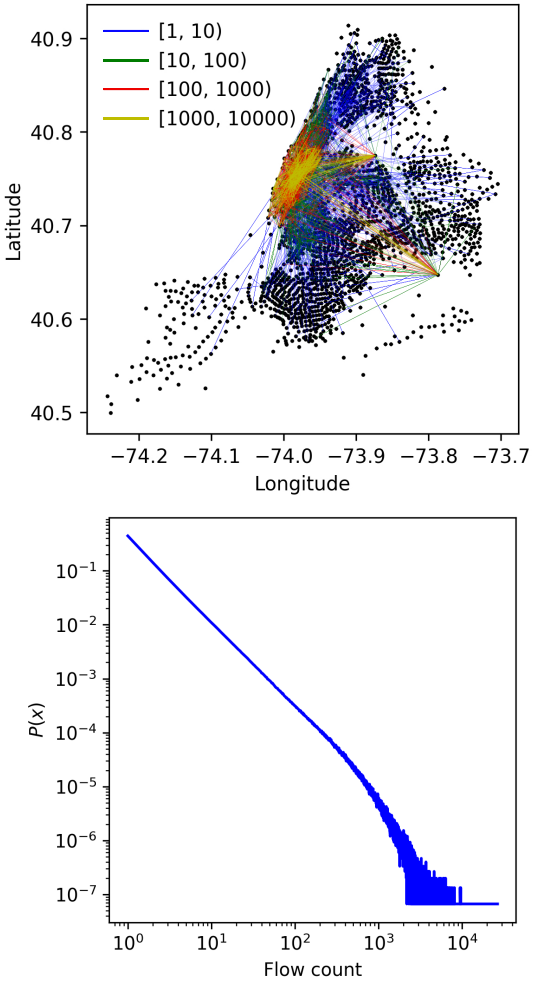

Fig. 3 Top: Spatial mapping of binned and randomly selected OD flows for taxi trips between New York City's census tracts in 2014 and 2015. The black dots represent New York City's census tracts and the colored lines represent the OD flow between them. Bottom: Probability distribution of all OD flows for Taxi trips between New York City's census tracts in 2014 and 2015.

given in table 3. Figure 4 shows the spatial distribution for three of the attributes: number of low income jobs, number of high income jobs, and number of amenities. For example, low-income jobs cluster in the Bronx, high-income jobs cluster in the Upper East Side and amenities cluster at Lower Manhattan. To sanity-check the variable selection method, a synthetic attribute is added to the dataset which represents a set of random samples from a standard normal distribution. For all job-related attributes the LEHD Origin-Destination Employment Statistics (LODES) data [51] is used which provides information to workforce dynamics. For the amenity

Table 3 Location attributes (urban features) related to attraction of trips, production of trips, and interaction between origin and destination.

Trip relation	Attribute (urban feature)
Destination (attraction)	No. of low-income residents No. of medium-income residents No. of high-income residents No. of low-income workers No. of medium-income workers No. of high-income workers No. of amenities
Origin (production)	No. of low-income residents No. of medium-income residents No. of high-income residents No. of low-income workers No. of medium-income workers No. of high-income workers No. of amenities
Intervening attributes	Distance between origin and destination No. of low-income residents closer to origin No. of medium-income residents closer to origin No. of high-income residents closer to origin No. of low-income workers closer to origin No. of medium-income workers closer to origin No. of high-income workers closer to origin No. of amenities closer to origin

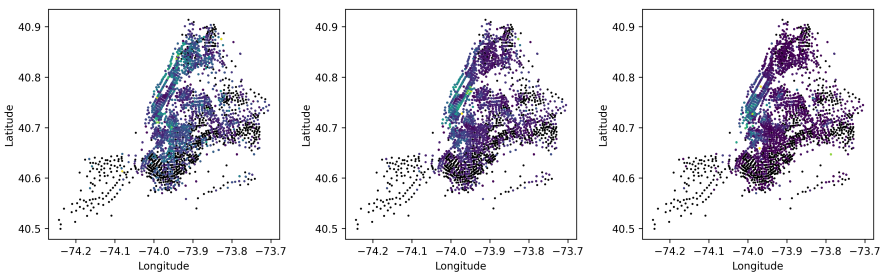

Fig. 4 Spatial mapping of the number of low income jobs of people living in each census tract (left), the number of high income jobs (middle), and the number of amenities (right). The dots represent census tracts and brighter colors correspond to higher numbers.

related attributes OpenStreetMap [52] is used. All Points-of-Interest that are declared as amenities have been included.

6. The flow data and the attribute data are merged into one data table with the following columns: ID of origin, ID of destination, year, weekday chunk, hour chunk, number of trips, and the 23 attributes associated with the origin and destination.

7. The data table is filtered for weekday chunk (e.g., only weekdays) and for hour chunk (e.g., only trips occurring between 7am and 9am). All trips where the origin equals the destination are filtered out.
8. The data table is split into a training set (all flows occurring in the year 2014) and a test set (all flows occurring in the year 2015).
9. From the training set a small subset is selected that is used as training data. This simulates the use case of limited access to training data. This is a likely scenario since it is difficult for urban planners to get access to data that covers the entire population (e.g., due to privacy concerns). The training data is then uniformly sampled into equally sized data bins until each bin is filled. The data bin edges are set to [1, 10, 100, 1000, 10000] and the bin size is set to 250. The training set therefore contains 1000 data points. The test data is treated in a similar fashion but additionally an unbinned test set is created to simulate a real world application. As such, the application is focused on the prediction (or 'generation') of flows based on urban features, without requiring the specific historical time series for a given OD pair.
10. Training and test sets are finally log-transformed and then linearly shifted to zero mean and scaled to one standard deviation.

3.2 Model configurations

The network depth was manually set and tested for $L \in \{0, 1, 3\}$, where $L = 0$ corresponds to the linear gravity model. Since all initial results were insensitive to the choice of the hyperparameter σ_g^2, it was set to 1 for all following experiments. The hyperparameters for the remaining priors are: (i) $\alpha_w = 1$ and $\beta_w = 10^{-2}$ which gives a weakly informative prior for σ_w^{-2}, (ii) $\alpha_b = 1$ and $\beta_b = 10^{-2}$ which gives a weakly informative prior for σ_b^{-2}, (iii) $\nu_0 = 1$ and $s_0 = 1$ which results in a weakly informative Student-$t_{\nu_0}(0, s_0^2)$ prior for the parameters far from zero, and (iv) $\alpha_n = 10^{-6}$ and $\beta_n = 10^{-6}$ which gives an uninformative prior for the noise σ_n^{-2}. As for the synthetic dataset, the HMC algorithm is used to find the best hyperparameter configuration of the Gaussian process kernel, in particular the variance of the global shrinkage parameter.

3.3 Results

The following metrics are used to quantify the prediction performance of the H-NNGP model:

- Mean squared error, MSE $= \frac{1}{N}\sum_{i=1}^{N}(y_i - z_i)^2$, where z_i denotes the predicted flow for an OD pair i, y_i is the actual flow, and N is the number of OD pairs.
- Mean absolute error, MAE $= \frac{1}{N}\sum_{i=1}^{N}|y_i - z_i|$. Large differences between predicted and actual flows are penalized less compared to the MSE.
- Mean absolute percentage error, MAPE $= \frac{1}{N}\sum_{i=1}^{N}|\frac{y_i - z_i}{y_i}|$.
- Common part of commuters, CPC $= \sum_{i=1}^{N}\frac{2\min(y_i, z_i)}{y_i + z_i}$. CPC is typically used in mobility flow modeling [7].

Table 4 shows the prediction performance results for three time periods (11pm - 1am, 7am - 9am, and 3pm - 5pm), for different data bins (flows between 10 - 100, flows between 100 - 1,000, flows between 1,000 - 10,000, all bins together, and an unbinned subset of flows) and for different model configurations ($L \in \{0, 1, 3\}$). The CPC values for all three time periods (here CPC is only computed for the unbinned data as is common practice in mobility modeling) show that the non-linear models ($L \in \{1, 3\}$) clearly outperform the linear model ($L = 0$). Further, for the non-linear models a higher number of hidden layers doesn't necessarily add to the prediction performance. This is also true for all other metrics for the unbinned data. We observe a similar picture for the binned data ('all') where the non-linear models outperform the linear ones. Compared to the unbinned results, the MSE and MAE are much higher and the MAPE is lower, which indicates the highly right-skewed nature of the flow data (with many small flows and a few very large flows). By zooming into the individual bins we can see the contributions of each bin to the overall errors. Generally, compared to the bins with smaller flows, the bins with larger flows contribute more to the MSE and MAE but contribute less to the MAPE. This shows that the models in absolute numbers are better in predicting small flows but in relative numbers are better in predicting large flows. The main reason is that the flows are log-transformed in the model. After prediction and inverse log-transforming, the absolute errors for large flows become dominant. With one exception, the non-linear models outperform the linear models for all individual bins. Interestingly, the errors for the period 11pm - 1am are much smaller than for the other two time periods. The higher accuracy might be due to the more specialized trips (such as taking a taxi back home from nightlife) where amenities cluster at certain locations and OD patterns are more predictable.

Figure 5 shows the absolute percentage error for the binned test data for the three different time periods and for the three different H-NNGP versions. The higher accuracy of the non-linear models compared to the linear gravity models is clearly visible, while there is no pronounced difference between one hidden layer and three hidden layers. For instance, the non-linear models seem to predict flows originating or ending at the airports (John F. Kennedy and LaGuardia) much better than the linear models.

To compare the prediction performance to existing approaches, we applied a deterministic neural network and a deterministic linear regression to our data. As is shown in table 5, H-NNGP outperforms the neural network in almost all cases. This may be expected since the small size of our training dataset (typical for urban planning applications) does not allow us to exploit the full potential of neural networks. Together with previous evaluations of different machine learning models [15, 19], this suggests that the accuracy of H-NNGP is competitive with state-of-the-art models that do not provide uncertainty estimates. H-NNGG also outperforms the linear regression in almost all cases (table 5).

Table 4 Prediction performance. The winning models are highlighted in bold.

Time	Bin	L	MSE	MAE	MAPE	CPC
11pm - 1am	10^1 - 10^2	0	13 913	33.8	1.21	-
		1	3 774	**28.1**	**0.97**	-
		3	**3 660**	28.2	0.98	-
	10^2 - 10^3	0	343 332	258.4	0.97	-
		1	86273	191.2	**0.77**	-
		3	**82 016**	**189.5**	**0.77**	-
	10^3 - 10^4	0	928 468	736.9	0.53	-
		1	347 263	443.4	0.31	-
		3	**326 449**	**426.2**	**0.30**	-
	all	0	321 473	258.4	1.41	-
		1	109 362	166.7	**0.99**	-
		3	**103 064**	**162.0**	**0.99**	-
	unbinned	0	24	2.5	1.72	0.51
		1	17	**1.9**	1.14	**0.60**
		3	**15**	**1.9**	**1.12**	**0.60**
7am - 9am	10^1 - 10^2	0	**16 656**	49.2	1.82	-
		1	22 438	**45.6**	**1.60**	-
		3	24 548	46.6	1.64	-
	10^2 - 10^3	0	419 559	285.9	0.91	-
		1	161 593	253.1	**0.89**	-
		3	**158 017**	**252.9**	0.90	-
	10^3 - 10^4	0	3 914 048	1280.0	0.67	-
		1	1 689 091	906.1	0.47	-
		3	**1 641 515**	**885.9**	**0.45**	-
	all	0	1 085 868	406.1	2.44	-
		1	467 710	302.6	**1.63**	-
		3	**455 499**	**297.7**	1.64	-
	unbinned	0	169	5.1	3.33	0.38
		1	**66**	**2.8**	**1.57**	**0.55**
		3	77	2.9	1.65	0.54
3pm - 5pm	10^1 - 10^2	0	14 147	43.6	1.73	-
		1	**9 247**	**35.6**	**1.34**	-
		3	9 467	36.0	1.35	-
	10^2 - 10^3	0	646 645	298.2	0.94	-
		1	128 027	217.2	**0.76**	-
		3	**124 870**	**216.0**	**0.76**	-
	10^3 - 10^4	0	3 434 681	1'208.0	0.71	-
		1	1 183 909	731.7	0.41	-
		3	**1 120 180**	**706.1**	**0.40**	-
	all	0	1 023 451	389.6	2.17	-
		1	330 176	247.4	**1.35**	-
		3	**313 529**	**240.8**	1.36	-
	unbinned	0	118	4.7	3.00	0.40
		1	**101**	**3.1**	**1.73**	**0.52**
		3	110	3.3	1.81	0.51

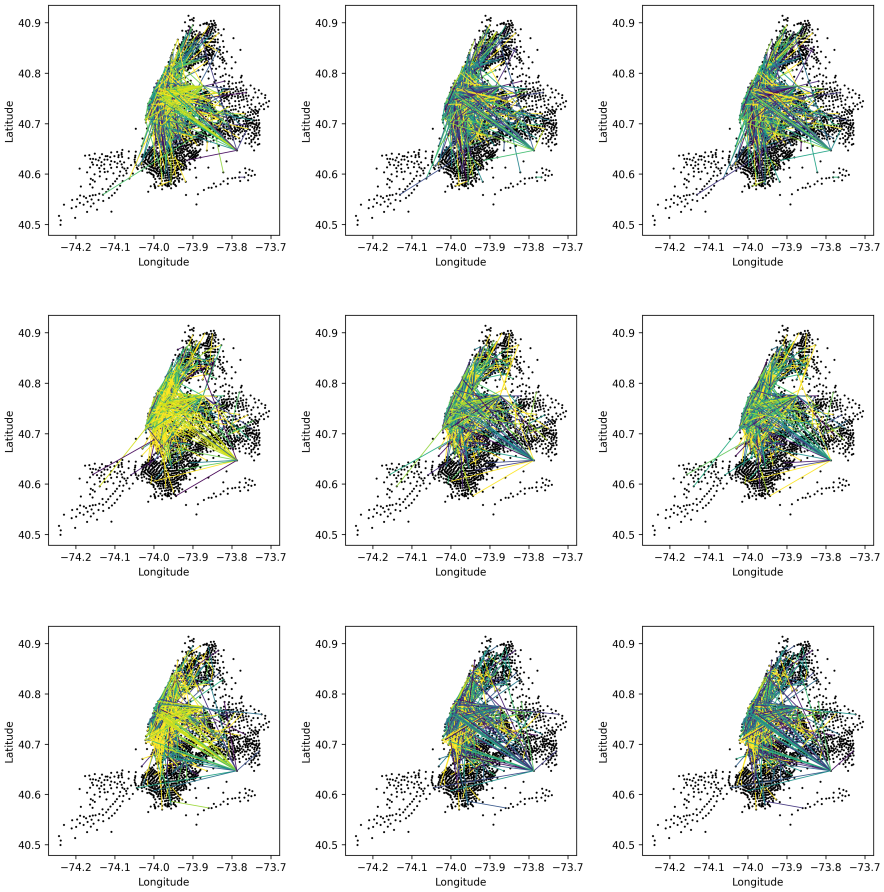

Fig. 5 Absolute percentage errors of the predicted flows in New York City for the binned test data. The errors are shown for flows occurring between 11pm - 1am (first row), 7am - 9am (second row), 3pm - 5pm (third row) and for the H-NNGP with zero (first column), one (second column), and three hidden layers (third column). The colormap is normalized within each row and brighter colors indicate larger errors.

The second part of results concerns the uncertainty estimation performance. An advantage of the Bayesian formulation compared to existing methods is that it allows to assign a variance to each test point prediction which encodes how certain the model is about a prediction. Figure 6 shows the correlation between uncertainty estimates and prediction error. For all cases there is a monotonic relationship between uncertainty estimates and prediction error: the more uncertain the model is about it's predictions (high predicted MSE) the larger the actual error will be (high actual MSE). This characteristic is a very valuable feature when the models are used in practice. For example, it allows urban planners to only rely on the model's predictions when the model is very certain. The non-linear models give much better estimates of the MSE compared to the linear models.

Table 5 Prediction performance of a deterministic neural network that has 1 hidden layer with 8 units and ReLU activation functions (NN), and a deterministic linear regression (LR). The performance metric is highlighted in bold in case one of these two baseline models performs better than the H-NNGP according to table 4.

Time	Bin	Model	MSE	MAE	MAPE	CPC
11pm - 1am	10^1 - 10^2	LR	10 005	37.7	1.35	-
		NN	5 914	35.2	1.15	-
	10^2 - 10^3	LR	263 685	251.1	0.88	-
		NN	88 264	205.8	0.83	-
	10^3 - 10^4	LR	875 771	720.5	0.53	-
		NN	473 782	519.5	0.37	-
	all	LR	287 421	254.0	1.52	-
		NN	142 194	192.0	1.44	-
	unbinned	LR	23	2.5	1.70	0.53
		NN	20	2.1	1.27	0.56
7am - 9am	10^1 - 10^2	LR	21'039	50.6	1.82	-
		NN	18 391	**45.0**	**1.57**	-
	10^2 - 10^3	LR	190 577	253.2	0.90	-
		NN	231 275	275.9	0.97	-
	10^3 - 10^4	LR	6 483 198	1 459.6	0.73	-
		NN	2 961 338	1198.2	0.60	-
	all	LR	1 667 564	442.5	2.51	-
		NN	800 101	380.6	1.89	-
	unbinned	LR	125	4.5	2.93	0.41
		NN	**38**	3.2	2.11	0.51
3pm - 5pm	10^1 - 10^2	LR	11 503	41.3	1.69	-
		NN	71 714	48.6	2.35	-
	10^2 - 10^3	LR	183 802	272.4	0.88	-
		NN	128 035	229.5	0.77	-
	10^3 - 10^4	LR	2 187 690	1092.3	0.64	-
		NN	1 626 269	927.7	0.55	-
	all	LR	593 904	353.0	2.00	-
		NN	455 106	302.4	1.65	-
	unbinned	LR	**87**	4.2	2.70	0.41
		NN	99	3.6	2.19	0.45

The third part of results concerns the variable selection performance. Figure 7 shows the posterior values for all local shrinkage parameters for the three versions of H-NNGP applied to the time period 7am - 9am (morning rush hours). The values are the means of the corresponding MCMC chains and indicate the importance of a particular feature. High relevance values in the non-linear models additionally indicate high non-linearity of the feature. The random feature is zero for all models showing that they pass this sanity check.

For the non-linear models, the variable selection results are the same for the model versions with 1 and 3 hidden layers, again suggesting that a larger

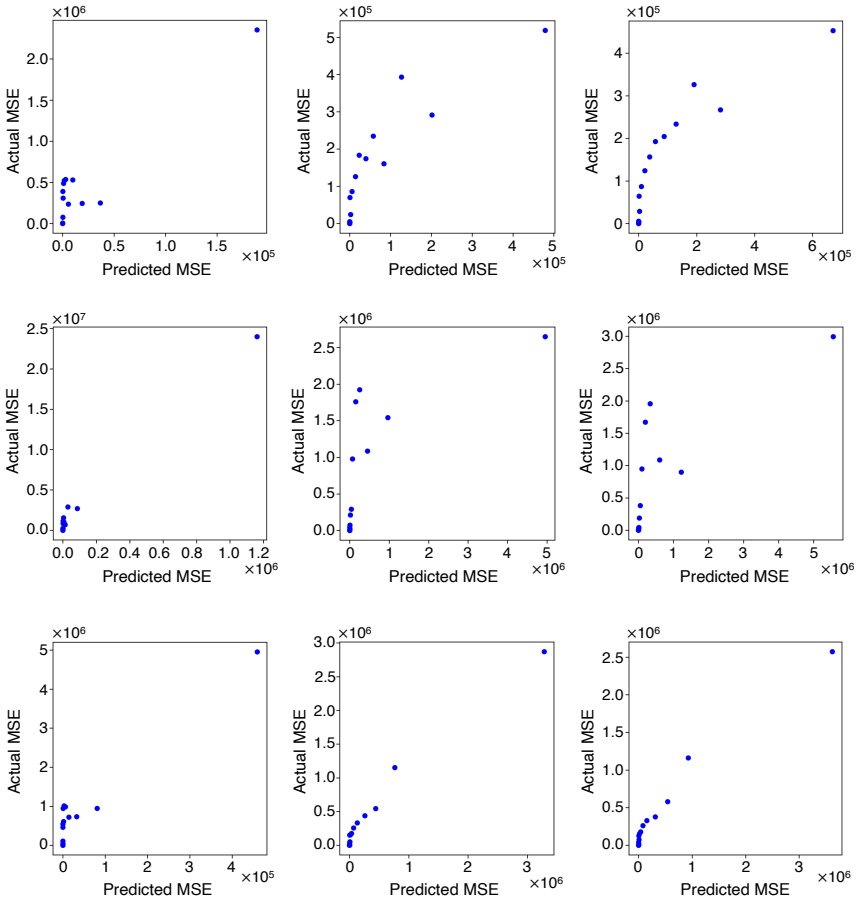

Fig. 6 Predicted versus actual MSE for the binned test data. Means are computed by averaging over 50 data points. The correlation plots are shown for flows occurring between 11pm - 1am (first row), 7am - 9am (second row), 3pm - 5pm (third row) and for the H-NNGP with zero hidden layer (first column), one hidden layer (second column), and three hidden layers (third column).

number of hidden layers do not necessarily discover more patterns in the data. The non-linear models detect the home-work trip patterns. For instance, at the origin location, the number of high-income residents is the most important feature for the taxi trips, while at the destination location the most important feature is the number of high-income workers. The relevant feature related to trip friction is distance. Because the non-linear models are able to pick up non-linear relations between the features, they also perform well at predicting trips that deviate from the home-work patterns. Examples are trips from and to the airports as has been shown in Fig. 5.

In summary, the comparison between the different versions of the H-NNGP model shows that i) the non-linear versions outperform the linear versions, ii) for the non-linear versions a higher number of hidden layers doesn't necessarily

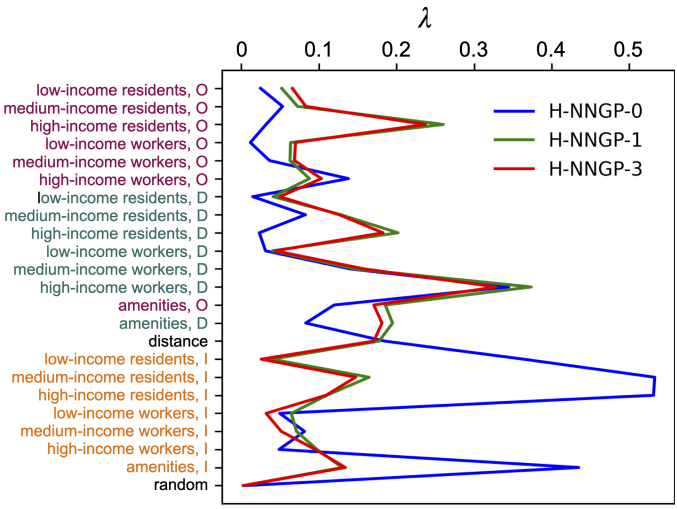

Fig. 7 Feature relevance for predicting flows that occurred 7am - 9am for the linear model with zero hidden layers (H-NNGP-0, equivalent to the linear gravity model), the non-linear model with one hidden layer (H-NNGP-1) and the non-linear model with three hidden layers (H-NNGP-3).

add to prediction performance (as such, we also expect similar performances with $L = 2$ hidden layers), iii) the more uncertain the models are about their predictions, the larger the actual error will be and iv) only the non-linear model versions can detect trips that diverge from predominant patterns and are related to more specialized places, such as trips from and to the airports.

4 Conclusion

This paper presents a novel machine learning approach for the prediction of mobility flows. The main advancement is the ability to quantify the uncertainty of the predictions, which so far has not been possible with existing approaches such as the traditional gravity model [48] or previous machine learning methods [20, 21]. This is achieved through fully Bayesian inference on deep neural networks that are formulated as Gaussian processes. The resulting H-NNGP model outperforms the prediction accuracy of traditional mobility models that do not consider non-linearity [48], and it allows for the identification of the most relevant urban features that affect the magnitudes of the mobility flows.

The added benefit of the uncertainty quantification is particularly relevant for the planning of transportation systems and other urban infrastructures, since such large-scale interventions require a high confidence in the mobility predictions. For instance, planners and policy makers may now conclude that there is a definitive need for a transport system extension only if the underlying model prediction (i.e., insufficient transport capacity relative to the predicted flow) also comes with a high certainty (i.e., low predicted MSE

value). This may help to reduce the risk of inadequate investments in the transport infrastructure.

Concerning the immediate application, it will be interesting to assess further the ideal size of the training dataset, below which the predictions become unstable and beyond which the computation becomes impractical. Moreover, to reduce the computational complexity and associated CPU times, future work should explore scalable Gaussian processes using approximation methods [37]. Finally, the identification of relevant geographic features that determine the flows can be taken as a starting point to gain a deeper understanding of the mechanisms behind observed mobility patterns. This may help to further refine existing explainable yet rather coarse-grained models with few parameters (e.g., [53]), so as to better capture location-specific variabilities. To that end, the proposed approach will need to be tested on more comprehensive mobility data (e.g., mobile phone records), on additional geographic features (e.g., efficiency of the transportation system), and it will need to be applied to different geographic regions across the world.

Acknowledgments. AS and MS acknowledge funding from the Singapore National Research Foundation (FI 370074016). MS acknowledges support from the National Research Programme 'Digital transformation' (NRP 77) of the Swiss National Science Foundation (project number 187443, entitled 'The role of digitalisation in the spatial distribution of the economy').

Authors' contributions. AS: conceptualization, methodology, data analysis, writing—initial draft. BSL: conceptualization, supervision, writing—reviewing and editing. MS: conceptualization, supervision, writing—final draft.

Conflict of interest. The authors declare no conflicts of interest.

References

[1] Batty, M., Axhausen, K.W., Giannotti, F., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., Ouzounis, G., Portugali, Y.: Smart cities of the future. The European Physical Journal Special Topics **214**, 481–518 (2012)

[2] Tirachini, A., Cats, O.: COVID-19 and public transportation: Current assessment, prospects, and research needs. Journal of Public Transportation **22**, 1–21 (2020)

[3] Redding, S.J., Turner, M.A.: Transportation costs and the spatial organization of economic activity. In: Duranton, G., Henderson, J.V., Strange, W.C. (eds.) Handbook of Regional and Urban Economics, pp. 1339–1398 (2015)

[4] Toole, J.L., Colak, S., Sturt, B., Alexander, L.P., Evsukoff, A., González, M.C.: The path most traveled: Travel demand estimation using big data resources. Transportation Research Part C **58**, 162–177 (2015)

[5] Schläpfer, M., Chew, H.J., Yean, S., Lee, B.-S.: Using mobility patterns for the planning of vehicle-to-grid infrastructures that support photovoltaics in cities (2021). Preprint at https://arxiv.org/abs/2112.15006

[6] Martinez-Cesena, E.A., Mancarella, P., Ndiaye, M., Schläpfer, M.: Using mobile phone data for electricity infrastructure planning. Conference on the Analyisis of Mobile Phone Networks (2015). Cambridge, USA

[7] Barbosa, H., Barthelemy, M., Ghoshal, G., James, C.R., Lenormand, M., Louail, T., Menezes, R., Ramasco, J.J., Simini, F., Tomasini, M.: Human mobility: Models and applications. Physics Reports **734**, 1–74 (2018)

[8] Zipf, G.K.: The P1P2/D hypothesis: on the intercity movement of persons. American Sociological Review **11**, 677–686 (1946)

[9] Wilson, A.G.: Urban and Regional Models in Geography and Planning. Wiley, New Jersey (1974)

[10] Erlander, S., Stewart, N.F.: The Gravity Model in Transportation Analysis: Theory and Extensions. CRC Press, Boca Raton (1990)

[11] Wilson, A.G.: A family of spatial interaction models, and associated developments. Environment and Planning A **3**, 1–32 (1971)

[12] Lenormand, M., Bassolas, A., Ramasco, J.J.: Systematic comparison of trip distribution laws and models. Journal of Transport Geography **51**, 158–169 (2016)

[13] Stouffer, S.A.: Intervening opportunities: A theory relating mobility and distance. American Sociological Review **5**, 845–867 (1940)

[14] Simini, F., González, M.C., Maritan, A., Barabási, A.-L.: A universal model for mobility and migration patterns. Nature **484**, 96 (2012)

[15] Pourebrahim, N., Sultana, S., Niakanlahiji, A., Thill, J.-C.: Trip distribution modeling with Twitter data. Computers, Environment and Urban Systems **77**, 101354 (2019)

[16] Robinson, C., Dilkina, B.: A machine learning approach to modeling human migration. In: Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies, pp. 1–8 (2018)

[17] Toqué, F., Côme, E., El Mahrsi, M.K., Oukhellou, L.: Forecasting dynamic public transport origin-destination matrices with long-short term memory recurrent neural networks. In: IEEE International Conference on Intelligent Transportation Systems (ITSC), pp. 1071–1076 (2016)

[18] Tillema, F., Van Zuilekom, K.M., Van Maarseveen, M.F.: Comparison of neural networks and gravity models in trip distribution. Computer-Aided Civil and Infrastructure Engineering **21**, 104–119 (2006)

[19] Yeghikyan, G., Opolka, F.L., Nanni, M., Lepri, B., Liò, P.: Learning mobility flows from urban features with spatial interaction models and neural networks. In: IEEE International Conference on Smart Computing, pp. 57–64 (2020)

[20] Luca, M., Barlacchi, G., Lepri, B., Pappalardo, L.: A survey on deep learning for human mobility. ACM Computing Surveys **55**, 1–44 (2021)

[21] Simini, F., Barlacchi, G., Luca, M., Pappalardo, L.: A Deep Gravity model for mobility flows generation. Nature Communications **12**, 1–13 (2021)

[22] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25**, 1097–1105 (2012)

[23] Graves, A.: Practical variational inference for neural networks. In: Advances in Neural Information Processing Systems, pp. 2348–2356 (2011)

[24] Murphy, K.P.: Machine Learning: A Probabilistic Perspective. MIT Press, Cambridge (2012)

[25] Steentoft, A.A.: Uncertainty quantification for complex systems: Application to the study of cities. PhD thesis, Nanyang Technological University, Singapore (2021)

[26] Gal, Y.: Uncertainty in deep learning. PhD thesis, University of Cambridge, Cambridge, UK (September 2016)

[27] Chen, X., He, Z., Sun, L.: A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation. Transportation Research Part C **98**, 73–84 (2019)

[28] Rodrigues, F., Pereira, F.C.: Beyond expectation: Deep joint mean and quantile regression for spatiotemporal problems. IEEE Transactions on Neural Networks and Learning Systems **31**, 5377–5389 (2020)

[29] Gammelli, D., Rodrigues, F.: Recurrent flow networks: A recurrent latent variable model for density estimation of urban mobility. Pattern Recognition **129**, 108752 (2022)

[30] Tygesen, M.N., Pereira, F.C., Rodrigues, F.: Unboxing the graph: Towards interpretable graph neural networks for transport prediction through neural relational inference. Transportation Research Part C **146**, 103946 (2023)

[31] Neal, R.M.: Bayesian learning for neural networks. PhD thesis, University of Toronto, Toronto, Canada (1995)

[32] Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., Rubin, D.B.: Bayesian Data Analysis. Chapman and Hall/CRC, New York (2013)

[33] Blundell, C., Cornebise, J., Kavukcuoglu, K., Wierstra, D.: Weight uncertainty in neural networks. arXiv preprint arXiv:1505.05424 (2015)

[34] Yao, Y., Vehtari, A., Simpson, D., Gelman, A.: Yes, but did it work?: Evaluating variational inference. arXiv preprint arXiv:1802.02538 (2018)

[35] Lee, J., Bahri, Y., Novak, R., Schoenholz, S.S., Pennington, J., Sohl-Dickstein, J.: Deep neural networks as Gaussian processes. In: International Conference on Learning Representations (2018)

[36] Matthews, A.G.d.G., Rowland, M., Hron, J., Turner, R.E., Ghahramani, Z.: Gaussian process behaviour in wide deep neural networks. arXiv preprint arXiv:1804.11271 (2018)

[37] Liu, H., Ong, Y.-S., Shen, X., Cai, J.: When Gaussian process meets big data: A review of scalable GPs. IEEE Transactions on Neural Networks and Learning Systems **31**, 4405–4423 (2020)

[38] Cho, Y., Saul, L.K.: Kernel methods for deep learning. In: Advances in Neural Information Processing Systems, pp. 342–350 (2009)

[39] Piironen, J., Vehtari, A., *et al.*: Sparsity information and regularization in the horseshoe and other shrinkage priors. Electronic Journal of Statistics **11**, 5018–5051 (2017)

[40] Carvalho, C.M., Polson, N.G., Scott, J.G.: Handling sparsity via the horseshoe. In: Artificial Intelligence and Statistics, pp. 73–80 (2009)

[41] Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)

[42] Betancourt, M., Girolami, M.: Hamiltonian Monte Carlo for hierarchical models. Current trends in Bayesian methodology with applications **79**, 2–4 (2015)

[43] Guyon, I.: Design of experiments of the NIPS 2003 variable selection benchmark. In: NIPS 2003 Workshop on Feature Extraction and Feature Selection, vol. 253 (2003)

[44] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge, Massachusetts (2016)

[45] Rifkin, R., Klautau, A.: In defense of one-vs-all classification. Journal of Machine Learning Research **5**, 101–141 (2004)

[46] Rifkin, R., Yeo, G., Poggio, T.: Regularized least-squares classification. Nato Science Series Sub Series III Computer and Systems Sciences **190**, 131–154 (2003)

[47] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015)

[48] Ortúzar, J.d.D., Willumsen, L.G.: Modelling Transport. John Wiley and Sons, Hoboken, New Jersey (2011)

[49] Census Tract Shapefiles. https://www.census.gov/cgi-bin/geo/shapefiles/index.php?year=2019&layergroup=Census+Tracts. Accessed: 2020-06-01

[50] TLC Trip Record Data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page. Accessed: 2020-06-01

[51] LEHD Employment Data. https://lehd.ces.census.gov/data/lodes. Accessed: 2020-07-05

[52] OpenStreetMap Data. https://www.openstreetmap.org/. Accessed: 2020-08-18

[53] Schläpfer, M., Dong, L., O'Keeffe, K., Santi, P., Szell, M., Salat, H., Anklesaria, S., Vazifeh, M., Ratti, C., West, G.B.: The universal visitation law of human mobility. Nature **593**, 522–527 (2021)