Variable Attention and Variable Noise: Forecasting User Activity

César Ojeda, Kostadin Cvejoski, Rafet Sifa, and Christian Bauckhage

Fraunhofer IAIS, Germany
{name.surname}@iais.fraunhofer.de

Abstract. The study of collective attention is of growing interest in an age where mass- and social media generate massive amounts of often short lived information. That is, the problem of understanding how particular ideas, news items, or memes grow and decline in popularity has become a central problem of the information age. Recent research efforts in this regard have mainly addressed methods and models which quantify the success of such memes and track their behavior over time. Surprisingly, however, the aggregate behavior of users over various news and social media platforms where this content originates has large been ignored even though the success of memes and messages is linked to the way users interact with web platforms. In this paper, we therefore present a novel framework that allows for studying the shifts of attention of whole populations related to websites or blogs. The framework is an extension of the Gaussian process methodology, where we incorporate regularization methods that improve prediction and model input dependent noise. We provide comparisons with traditional Gaussian process and show improved results. Our study in a real world data set, uncovers hidden patterns of user behavior.

Keywords: Gaussian process, Regularization Methods

1 Introduction

Over the last couple of years, so called "question answering" (QA) sites have gained considerable popularity. These are internet platforms where users pose questions to a general population. Yahoo Answers, Quora and the Stack Exchange family establish internet communities which provide natural and seamless ways for organizing and providing knowledge [1]. So far, dynamical aspects of such questions answering sites have been studied in different contexts. Previous work in this area includes studying causality aspects through quasi experimental designs [8], user churn analysis through classification algorithms such as support vector machines or random forests [9], and predictions of the future value of questions answers pairs according to the initial activity of the question post [2]. In contrast to previous work where long term activity of users is being predicted, our focus in this paper is time series analysis related to user-defined tags. This

approach allows detailed daily analysis of the behavior of users and we concentrate on the QA site Stackoverflow. This platform has an established reputation on the web and boasts a community of over 5 million distinct active users who, so far, have provided more 18 million answers to more than 11 million questions. Thanks to the shear size of the corresponding data set as well as because of the regular activity of the user base, we are able to mine temporal data in order to uncover defining aspects of the dynamics of the user behavior.

Due to the complexity of user-system interaction (millions of people discuss thousands of topics), flexible and accurate models are required in order to guarantee reliable forecasting. In recent years the Bayesian setting and the Gaussian Process (GP) framework [11, 5] has shown to provide an accurate and flexible tool for time series analysis. In particular, the possibility of incorporating error ranges as well as different models with the selection of different kernels, permits interpretability of the results. In this work, we model changes in attention as a variability in the fluctuation of the time series of occurrences of user defined tags which can be categorized as a special case of *heterocedasticity* or input dependent noise. We provide an extension of sparse input Gaussian Processes [15, 14] which allow us to model functional dependence in the time variation of the fluctuations. In practical experiments, we study the top 10 different tags for the Stackoverflow data set over different years, spanning a data set of over 2.9 million questions. We find that our model outperform predictions made by the simple GP model under variable noise. In particular, we uncover weekly and seasonal periodicity patterns as well as random behavior in monthly trends. All in all, we are able to forecast the number of questions within a 5 percent error 20 days in the future.

In the next section, we formally introduce the Gaussian Process framework and provide details regarding our extensions towards variable noise models. We then show an analysis of the periodicity of the time series of tag activity as apparent from the Stackoverflow data set. Next, we compare our prediction results with those of other models and discuss the advantages of introducing functional dependencies on noise terms. Finally, we provide conclusions and directions of future work.

2 A Model for Time Series Analysis

In this section, we propose a Gaussian process (GP) model for regression that extends the sparse pseudo-input Gaussian process (SPGP) for regression [14]. Our model deals with the problem of over fitting that hampers the SPGP model and makes it possible to analyze the function of the uncertainty added to every pseudo-input. Analyzing the uncertainty function, we indirectly analyze the effects of heteroscedastic noise.

A GP is a Bayesian model that is commonly used for regression tasks [11]. The main advantages of this method are its non-parametric nature, the possibility

of interpreting the model through flexible kernel selection, and the confidence intervals (error bars) obtained for every prediction. The non-parametric nature of this method has a drawback, though. The computational cost of the training is $\mathcal{O}\left(N^3\right)$, where N is the number of training points. There are many sparse approximation methods of the full GP that try to lower the computational cost of the training to $\mathcal{O}\left(M^2 N\right)$ where M is the size of the subset of the training points that are used for approximation (i.e. the active set) and typically $M \ll N$ [13, 12]. The M points for the approximation are chosen according to various information criteria. This leads to difficulties w.r.t. learning the kernel hyperparameters by maximizing the marginal likelihood of the GP using gradient ascent. The re-selection of the active set causes non-smooth fluctuations of the gradients of the marginal likelihood, which results likely convergence to suboptimal local maxima [14].

2.1 Gaussian Process for Regression

Next, we first briefly review the GP model for regression, yet, for a detailed discussion we refer to [11, 10].

Consider a data set $\mathcal{D}$ of size N containing pairs of input vectors $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$ and real value target points $\mathbf{y} = \{y_n\}_{n=1}^N$. In order to apply the GP model to regression problems, we need to account for noise in the available target values, which are thus expressed as

$$y_n = f_n + \epsilon_n \tag{1}$$

where $f_n = f(x_n)$ and ϵ_n is a random noise variable which is independently chosen for each n. We shall consider a noise process following a Gaussian distribution defined as

$$p(\mathbf{y} \mid \mathbf{f}) = \mathcal{N}\left(\mathbf{y} \mid \mathbf{f}, \sigma^2 \mathbf{I}\right) \tag{2}$$

where $\mathcal{N}(\mathbf{y} \mid \mathbf{m}, \mathbf{C})$ is a Gaussian distribution with mean $\mathbf{m}$ and covariance $\mathbf{C}$. The marginal distribution of $p(\mathbf{f})$ is then given by another Gaussian distribution, namely $p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f} \mid \mathbf{0}, \mathbf{K}_N)$. The covariance function that determines $\mathbf{K}_N$ is chosen to express the property that, if points $\mathbf{x}_n$ and $\mathbf{x}_m$ are similar, the value $[\mathbf{K}_N]_{nm}$ should express this similarity. Usually, this property of the covariance function is controlled by small number of hyperparameters $\boldsymbol{\theta}$. Integrating out over $\mathbf{f}$, we obtain the marginal likelihood as

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) &= \int p(\mathbf{y} \mid \mathbf{f})\, p(\mathbf{f} \mid \mathbf{X}, \boldsymbol{\theta}) \\ &= \mathcal{N}\left(\mathbf{y} \mid \mathbf{0}, \mathbf{K}_N + \sigma^2 \mathbf{I}_N\right), \end{aligned} \tag{3}$$

which is used for training the GP model by maximizing it with respect to $\boldsymbol{\theta}$ and σ^2. The distribution of the target value of a new point $\mathbf{x}$ will then be

$$\begin{aligned} p(y \mid \mathbf{x}, \mathcal{D}, \boldsymbol{\theta}) = \mathcal{N}\Big(y \mid \mathbf{k_x}^T \left(\mathbf{K}_N + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y}, K_{\mathbf{xx}} \\ -\mathbf{k_x}^T \left(\mathbf{K}_N + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{k_x} + \sigma^2\Big), \end{aligned} \tag{4}$$

where $[\mathbf{k}_{\mathbf{x}}]_n = K(\mathbf{x}_n, \mathbf{x})$ and $K_{\mathbf{xx}} = K(\mathbf{x}, \mathbf{x})$. In order to predict with GP model, we need to have all the training data available during run-time, which is why the GP for regression is referred to as a non-parametric model.

2.2 SPGP and SPGP+HS Models

An approximation of the full GP model for regression is presented in [14] in which the authors propose the sparse pseudo-input Gaussian process (SPGP) regression model that enables a search for the kernel hyper-parameters and the active set in a single joint optimization process. This is possible because it is allowed for the active set (pseudo-inputs M) to take any position in the data space, not only to be a subset of the training data. Parameterizing the covariance function of the GP by the pseudo-inputs, gives the possibility for learning the pseudo-inputs using gradient ascent. This is a major advantage, because it improves the model fit by fine tuning the locations of the pseudo-inputs. Let, $\overline{\mathbf{X}} = \{\overline{\mathbf{x}}_m\}_{m=1}^M$ be the pseudo-inputs and $\overline{\mathbf{f}} = \{\overline{f}_m\}_{m=1}^M$ are the pseudo targets, the predictive distribution of the model for a new input $\mathbf{x}_*$ will be given by

$$\begin{aligned} p\left(y_* \mid \mathbf{x}_*, \mathcal{D}, \overline{\mathbf{X}}\right) &= \int p\left(y_* \mid \mathbf{x}_*, \overline{\mathbf{X}}, \overline{\mathbf{f}}\right) p\left(\overline{\mathbf{f}} \mid \mathcal{D}, \overline{\mathbf{X}}\right) d\overline{\mathbf{f}} \\ &= \mathcal{N}\left(y_* \mid \mu_*, \sigma_*^2\right), \\ \mu_* &= \mathbf{k}_*^\top \mathbf{Q}_M^{-1} \mathbf{K}_{MN} \left(\mathbf{\Lambda} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y} \\ \sigma_*^2 &= K_{**} - \mathbf{k}_*^\top \left(\mathbf{K}_M^{-1} - \mathbf{Q}_M^{-1}\right) \mathbf{k}_* + \sigma^2, \end{aligned} \tag{5}$$

where $\mathbf{K}_N$ is the covariance matrix of the training data, $\mathbf{K}_M$ is the covariance matrix of the pseudo inputs, σ^2 is the noise, $\mathbf{Q}$ is defined as

$$\mathbf{Q} = \mathbf{K}_{MN} \left(\mathbf{\Lambda} + \sigma^2\right)^{-1} \mathbf{K}_{NM} \tag{6}$$

and $\mathbf{\Lambda}$ is defined as

$$\mathbf{\Lambda} = \operatorname{diag}\left(\mathbf{K}_N - \mathbf{K}_{NM} \mathbf{K}_M^{-1} \mathbf{K}_{MN}\right). \tag{7}$$

Finding the pseudo input locations $\overline{\mathbf{X}}$ and the hyperparameters (kernel parameters and noise) $\mathbf{\Theta} = \{\boldsymbol{\theta}, \sigma^2\}$ can be done by maximizing the marginal likelihood (8) with respect to the parameters $\{\overline{\mathbf{X}}, \boldsymbol{\Theta}\}$.

$$\begin{aligned} p\left(\mathbf{y} \mid \mathbf{X}, \overline{\mathbf{X}}, \mathbf{\Theta}\right) &= \int p\left(\mathbf{y} \mid \mathbf{X}, \overline{\mathbf{X}}, \overline{\mathbf{f}}\right) p\left(\overline{\mathbf{f}} \mid \overline{\mathbf{X}}\right) d\overline{\mathbf{f}} \\ &= \mathcal{N}\left(\mathbf{y} \mid 0, \mathbf{K}_{NM} \mathbf{K}_M^{-1} \mathbf{K}_{MN} + \mathbf{\Lambda} + \sigma^2 \mathbf{I}\right) \end{aligned} \tag{8}$$

One positive effect of the sparsity of the SPGP model is the capability of learning data sets that have variable noise where the term *variable noise* refers to noise which depends on the input. However, it is important to note, that this capability is limited and an improvement of the SPGP model is presented in

[15]. Introducing an additional uncertainty h_m parameter to every pseudo-input point makes the model more flexible and allows for improved representations of heteroscedastic data sets. The covariance matrix of the pseudo-inputs is defined by

$$\mathbf{K}_M \rightarrow \mathbf{K}_M + \mathtt{diag}\left(\mathbf{h}\right), \tag{9}$$

where $\mathbf{h}$ is a positive vector of uncertainties that needs to be learned and $\mathtt{diag}\left(\mathbf{h}\right)$ represents a diagonal matrix whose elements are those of the $\mathbf{h}$ vector. This extension allows the possibility of gradual influence on the pseudo inputs. This means that if uncertainty $h_m = 0$, then the pseudo input m behaves like in the standard SPGP. Yet, as h_m grows the particular pseudo input has less influence on the predictive distribution. This possibility of partially turning off the pseudo inputs allows a larger noise variance in the prediction. The authors of [15] refer to this as heteroscedastic (input dependable noise) extension SPGP+HS.

2.3 SPGP+FUNC-HS

Introducing the heteroscedastic extension to the SPGP empowers the model to learn from data sets with varying noise. However, making the model this flexible may cause problems of over fitting. Also, using the SPGP+HS to predict user and website activities, does not allow us to interpret the behavior of the noise because noise is represented as a positive vector $\mathbf{h}$ of uncertainties and attempts of interpreting these values do not yield meaningful information about the behavior of the noise.

One way of solving the problems of over fitting and lack of interpretability will be to put a prior distribution over the vector $\mathbf{h}$ of uncertainties. However, taking this approach leads to computationally intractable integrals.

The solution which we propose for these problems is to make use of an uncertainty function that depends on the pseudo-inputs. Our covariance function of the pseudo-inputs is defined as

$$\mathbf{K}_M \rightarrow \mathbf{K}_M + \mathtt{diag}\left(f_h\left(\overline{\mathbf{x}}_m\right)\right), \tag{10}$$

where f_h is the uncertainty function and $\overline{\mathbf{x}}_m$ is a pseudo-input. By defining the heteroscedastic extension in this way, it is possible for the parameters of the uncertainty function to be learned by the gradient based maximum likelihood approaches. Hence, later on, we are able to interpret the parameters of the heteroscedastic noise function as parameters that govern the noise in the model. Another advantage of having a heteroscedastic function is that it restricts the parameter search space when learning the model. This restriction can be beneficial when learning the model, because, it removes unnecessary local maxima. This results in much faster convergence when learning the model and also in improved chances of reducing over fitting. In the following, we will refer to our new heteroscedastic function model as **SPGP+FUNC-HS**.

For modeling the Stackoverflow data set, we introduce two heteroscedastic noise functions. In general, we may use any function that can describe the noise of the given data set. The first heteroscedastic noise function which we consider is the simple sine function defined by

$$f_h\left(\overline{\mathbf{x}}_m\right) = a \sin\left(2\pi\omega\overline{\mathbf{x}}_m + \varphi\right), \tag{11}$$

where a is the amplitude, ω is the frequency and φ is the phase. We refer to this model as **SPGP+SIN-HS**. The second heteroscedastic noise function we investigate is a product of the sine function and an RBF kernel, namely

$$f_h\left(\overline{\mathbf{x}}_m, \mathbf{h}_m\right) = c^2 e^{-\frac{(\overline{\mathbf{x}}_m - \mathbf{h}_m)^2}{2l^2}} \sin\left(2\pi\omega\overline{\mathbf{x}}_m + \varphi\right), \tag{12}$$

where c is the variance, $\mathbf{h}_m$ is a mean associated with every pseudo-input $\overline{\mathbf{x}}_m$ in the RBF kernel, and l is the length scale of the RBF kernel. The mean in the RBF kernel can be initialized at random or set by the user if the user has corresponding prior knowledge. Setting a mean for every pseudo-input point divides the whole input space into regions where, in each region, we have a function governing the uncertainty associated with every pseudo input. The uncertainty function defined like this then behaves like mixture of experts and we refer to this model as **SPGP+RBFSIN-HS** model.

3 Results

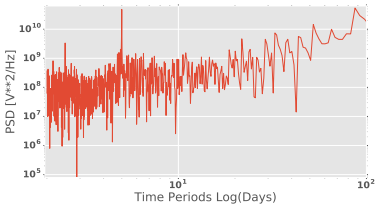


Fig. 1: Spectral Density Estimation on the Stackoverflow dataset using a periodogram. We observe two peaks at two and a half and five days, where the latter peak is the doubled period of the former peak.

In the previous section, we presented the Gaussian process method and two extensions of this method, the SPGP+HS and the SPGP+FUNC-HS. In this section, we present results we obtained when using these models on our Stackoverflow data set.

In order to test our models, we used publicly available data-dumps of Stackoverflow[1]. The data set contains the number of questions and answers of postings

[1] Downloadable URL: www.archive.org/details/stackexchange

	MSE				NLPD				NLML			
	GP	RBFSIN	HS	SIN	GP	RBFSIN	HS	SIN	GP	RBFSIN	HS	SIN
android	960.88	**692.03**	887.45	720.75	4.65	**4.49**	4.61	**4.49**	-1076.37	**-948.40**	-1149.22	-993.23
c#	1029.06	**881.11**	950.64	894.61	4.70	**4.62**	4.64	**4.62**	-1003.23	**-949.54**	-962.43	-961.62
c++	1216.94	**533.68**	5068.20	675.84	4.84	**4.45**	6.02	4.66	-717.14	**-698.50**	-756.95	-716.58
html	681.57	**678.19**	774.17	754.95	4.47	**4.45**	4.51	4.50	-841.93	**-784.78**	-798.28	-820.60
ios	2598.35	**1474.72**	3064.63	1660.90	5.82	**4.81**	5.53	4.86	-757.36	**-737.24**	-750.69	-740.49
java	1917.86	**1431.70**	3446.30	1782.17	5.12	**4.90**	5.79	4.95	-1098.13	**-1034.83**	-1087.29	-1068.30
javascript	2992.30	**1869.61**	2396.68	2102.05	6.09	**4.97**	5.52	5.22	-1493.42	**-883.31**	-1054.49	-1044.76
jquery	**808.26**	825.28	989.07	1163.88	**4.57**	4.77	4.69	4.73	-957.31	-932.99	**-866.17**	-862.45
php	5892.26	**907.07**	5379.89	2745.40	6.83	**4.60**	6.13	5.15	-1042.95	-883.65	-945.85	**-853.21**
python	**604.89**	702.25	744.28	881.65	**4.44**	4.58	4.53	4.62	**-782.68**	-842.76	-787.24	-788.14

Table 1: Results showing the MSE and NLPD (smaller better) on the 2014 question test set and NLML (larger is better) on the 2014 question training set. GP indicates a pure Gaussian process, HS indicates a sparse pseudo-input Gaussian process with heteroscedastic noise, SIN-HS refers to a sparse pseudo-input Gaussian process with sine functional noise, and RBFSIN-HS refers to a sparse pseudo-input Gaussian process with sine and RBF kernel functional noise.

classified by tag for every business day. The models are trained on a data set containing information about daily postings in the time between 01.02.2014 to 31.08.2014. The evaluation of the models is done on a test set containing postings for the first 21 working days in September 2014.

The performance of the presented models depends on the choice of the kernels used for the covariance matrix. When working with GPs, an additional analysis is required to select proper kernels for the covariance matrix. Because we work with a data set that reflects user behavior, we supposed that it may show a form of periodicity in the behavior of the users. Accordingly, we performed *spectral density estimation* analysis [6, 4, 7] of the time series using a *periodogram* analysis [6, 4, 7]. This analysis shows the power (amplitude) of the time series as a function of frequency and in this way we are able to verify if there are indeed periodicities and, if so, at what frequency they occur.

A periodogram of the time series data that we are analyzing is shown in Figure 1. Since all our tag related time series tag have almost the same periodogram, we only show one of them. For better interpretability we converted the frequencies into periods to observe in how many days the periods occur. There are two apparent peaks, the first occurring at two and a half days and the second at five days. In this case the period of five days appears as an echo of the two and a half days period, therefore we dismiss the second period and we only take into account the first period. Additional characteristics of this data set are minor irregularities and a long term rising trend in the overall time series.

Given these observations, our models that show the best performance as a covariance function use a sum of four kernels

$$k(x, x\prime) = k_1(x, x\prime) + k_2(x, x\prime) + k_3(x, x\prime) + k_4(x, x\prime). \tag{13}$$

The question of how to choose these kernels and the particular role of each kernel in the learned model will be discussed in the next section.

Next, we present the result achieved for the top ten tags according to the number of posted questions and answers in the 2014 Stackoverflow data set. Table 1 presents the results of the different models of the posted questions time series and Table 2 presents the results of the different models for the posted answers time series. In order to compare the prediction models, we considered the following measures:

- **Mean Square Error (MSE)** to accounts for the accuracy of the prediction of an unseen data point
- **Negative Log Predictive Distribution (NLPD)** to obtain a confidence for the predicted values on an unseen data point
- **Negative Log Marginal Likelihood (NLML)** to account for how well the model fits the training data.

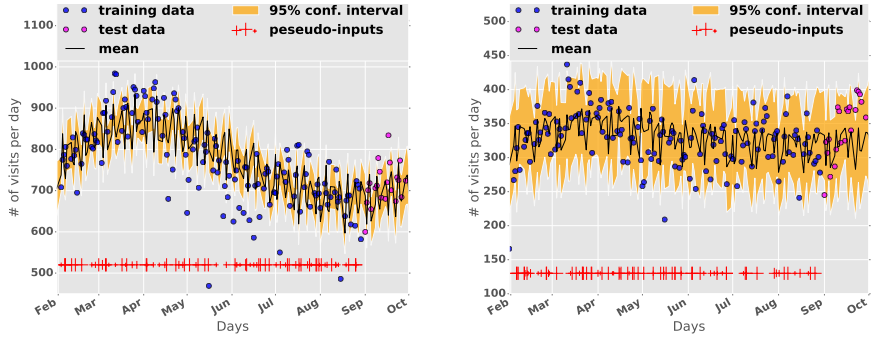
	MSE				NLPD				NLML			
	GP	RBFSIN	HS	SIN	GP	RBFSIN	HS	SIN	GP	RBFSIN	HS	SIN
android	1097.05	1098.29	1041.58	**1031.10**	4.80	4.79	4.81	**4.78**	**-903.82**	-919.78	-913.40	-927.61
c#	2889.76	**2723.95**	2998.26	2878.46	5.24	**5.18**	5.24	5.22	-1007.62	**-983.75**	-989.81	-995.13
c++	1602.27	**1436.71**	3491.81	3010.85	4.89	**4.85**	6.21	5.15	-825.76	-805.98	-886.82	**-775.62**
html	**1856.82**	2016.96	2162.96	1904.25	4.98	4.99	5.02	**4.96**	-1082.09	-957.67	**-907.46**	-954.44
ios	3944.90	**1541.55**	5156.82	5017.53	5.74	**4.87**	5.48	5.41	-831.93	-839.15	-778.98	**-777.68**
java	3207.22	**2987.25**	4085.50	3090.13	5.38	**5.19**	5.35	5.20	-1283.56	**-1016.72**	-1024.00	-1047.48
javascript	5360.20	**4869.97**	5434.37	5374.24	5.61	**5.50**	5.68	5.51	-1141.66	**-1110.28**	-1139.14	-1131.77
jquery	1817.16	1728.42	1749.74	**1725.81**	5.12	5.03	5.07	**5.00**	**-976.82**	-1021.99	-1009.31	-1023.81
php	2950.13	**2948.65**	3076.88	2982.74	**5.16**	**5.16**	5.20	5.17	-1011.84	-1015.56	-995.81	**-994.36**
python	911.70	606.00	1660.13	**605.22**	**4.64**	**4.64**	4.90	**4.64**	-867.67	-820.73	**-792.96**	-813.29

Table 2: Results showing the MSE and NLPD (smaller is better) on the 2014 answers test set and NLML (larger is better) on the 2014 answers training set. GP indicates a pure Gaussian process, HS indicates a sparse pseudo-input Gaussian process with heteroscedastic noise, SIN-HS refers to a sparse pseudo-input Gaussian process with sine functional noise, and RBFSIN-HS refers to a sparse pseudo-input Gaussian process with sine and RBF kernel functional noise.

For the MSE and the NLPD measures, smaller values are better, and for the NLML larger values are better. The best model for each tag has been chosen using the Akaike information criterion (AIC) [3]. We observe that models with functional noise perform better in nine of the ten tags in the answer time series, and eight of the ten tags in the question time series. The superior performance of the SPGP+FUNC-HS over the full GP can be attributed to the fact that the data set contains variable noise. Note that for this data set, SPGP+FUNC-HS performs better, because of the sparsity of the model and the additional functional noise that is added to the pseudo-inputs. SPGP+HS performs worse than the best models because, adding only a positive vector of uncertainty increases

the flexibility of the covariance function, which at the end can lead to over fitting and convergence to bad local maxima. Using a functional noise constraint, the optimization space shrinks and implicitly prunes bad local maximas. The drawback is that the function of the noise should follow the distribution of the noise in the data set, otherwise the model will perform poorly. This is probably the case why the SPGP+FUNC-HS performs worse on one tag for the answers and on two tags for the questions.

In Fig. 2, we present two learned models, one for the tag "Java" (Fig. 2a) and one for the tag "iOS" (Fig. 2b). We observe that the model that models the Java tag strives to predict the test point using the mean. In contrast, the model that models the iOS tag predicts the test points in terms of noise.



(a) "Java" (b) "iOS"

Fig. 2: Models learned with SPGP+SIN-HS for the tags "Java" and "iOS" in the 2014 data set.

4 Analysis

The different kernels in Eq. (13) allow us to dissect the dynamical behavior of the population w.r.t. different scales and patterns. In order to portrait these behaviors, we calculated the mean function and variance Eq. (5) by generating vector $\mathbf{k}_*^\top$ using independent kernels. We present the values of each kernel in the "android" question data set in Fig. 3

– **Mean trends** (Fig. 3a) characterize the behavior of the population of users over scales measured in months and represent the global mean behavior of the population. We hypothesize that they are driven by the shear size of the user base. The more people interested in the tag are visiting the site, the higher the average number of questions per month. Further, this overall trend might

represent the changes in the dominance of this particular tag of questions in the data set. Because the tag refers to a programming language, trends like this indicate changes in attention to various languages. Such dynamics are modeled using the rational quadratic kernel

$$k_1(x, x') = \theta_6^2 \Big(1 + \frac{(x - x')^2}{2\theta_8 \theta_7^2}\Big)^{-\theta_8} \tag{14}$$

– **Seasonal trends** (Fig. 3b) arise on a time scale smaller than major trends and show both periodical and stochastic patterns. They represent changes in the population behavior throughout the different months of the year which can be uncovered with the Ornstein-Ulenbeck kernel

$$k_2(x, x') = \theta_1 \exp(-\frac{|x - x'|}{\theta_2}). \tag{15}$$

– **Weekly periods** (Fig. 3c) as obtained from the periodogram represent weekly usage patterns and fine grained periods of activity in our data set. We hypothesize that such behaviors are related natural work patterns during the working week and model them using the following kernel (16).

$$k_3(x, x') = \theta_3^2 \exp\Big(L_1 + L_2\Big) \tag{16}$$

where we define L_1 and L_2 as

$$L_1 = -\frac{(x - x')^2}{2\theta_4^2} \tag{17}$$

and

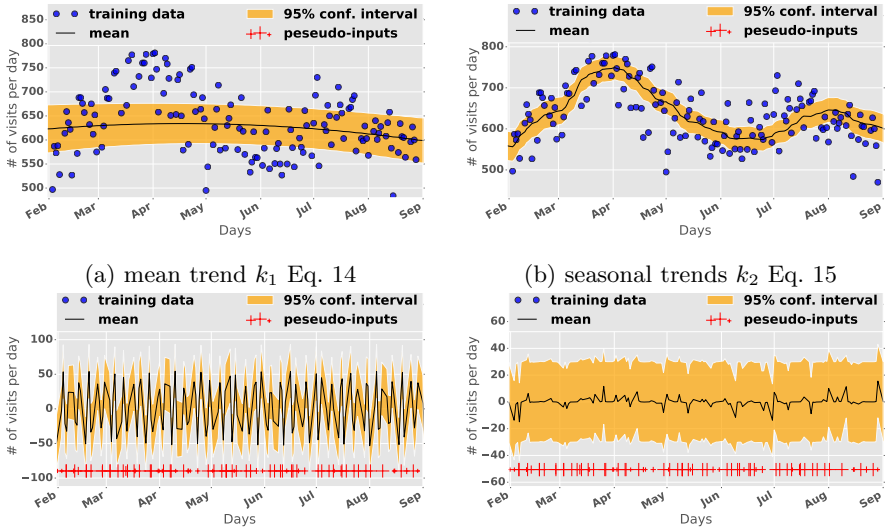
$$L_2 = -\frac{2\sin^2[\pi(x - x')/P]}{\theta_5^2}. \tag{18}$$

– **Weekly noise** (Fig. 3d) are fluctuations in the weakly behavior which can be expected due to the statistical nature of our data set. Randomness in the behavioral pattern of each user might give rise to fluctuations which we model using the following kernel

$$k_4(x, x') = \theta_9^2 \Big(1 + \frac{(x - x')^2}{2\theta_{11}\theta_{10}^2}\Big)^{-\theta_{11}} \tag{19}$$

5 Conclusion and Future Work

In this paper, we addressed the problem of forecasting the daily posting behavior of users of the Stackoverflow question answering web platform. In order to accomplish this task, we extended the variable noise pseudo inputs Gaussian



(a) mean trend k_1 Eq. 14

(b) seasonal trends k_2 Eq. 15

(c) weekly periods k_3 Eq. 16

(d) weekly noise k_4 Eq. 19

Fig. 3: Decomposition of the SPGP+SIN-HS model for the "android" tag in different kernels. We observe four main behaviors: mean trends, seasonal trends, weekly periods, and weekly noise.

Process framework by introducing a functional noise variant. The idea of using functional descriptions of noise allowed us to study periodic patterns in collective attention shifts and was found to act as a regularizer in model training.

Our extended Gaussian Process framework with functional representations of various kinds of noise provides the added advantage of increased interpretability of results as the different kernels defined for this purpose can uncover different kinds of dynamics. In particular, our kernels revealed major distinct characteristics of the question answering behavior of users. First of all, there are major trends on time scales of about six months showing growing and declining interest in particular topics or corresponding tags. Second of all, these major trends are perturbed by seasonal behavior, for example overall activities usually drop during the summer season. Third of all, on a fine grained scale, there are weekly patterns characterized by periods of 2.5 days. Fourth of all, there are noisy fluctuations in activities on daily scales.

Given the models and results presented in this paper, there various directions for future work. First and foremost, we are currently working on implementing a distributed Gaussian Process framework in order to extend our approach towards massive amounts of behavioral data (use of tags, comments, and likes) that can be retrieved from similar social media platforms such as Twitter or Facebook.

References

1. L. A. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman. Knowledge Sharing and Yahoo Answers: Everyone Knows Something. In *Proc. of ACM WWW*, 2008.
2. A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Discovering Value from Community Activity on Focused Question Answering Sites: A Case Study of Stack Overflow. In *Proc. of ACM KDD*, 2012.
3. C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
4. J. D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.
5. K. Kersting, C. Plagemann, P. Pfaff, and W. Burgard. Most likely heteroscedastic gaussian process regression. In *Proceedings of the 24th international conference on Machine learning*, pages 393–400. ACM, 2007.
6. D. G. Manolakis, V. K. Ingle, and S. M. Kogon. *Statistical and Adaptive Signal Processing: Spectral Estimation, Signal Modeling, Adaptive Filtering, and Array Processing*. Artech House Norwood, 2005.
7. C. Ojeda, R. Sifa, and C. Bauckhage. Investigating and Forecasting User Activities in Newsblogs: A Study of Seasonality, Volatility and Attention Burst. *Work On Progress*, 2016.
8. H. Oktay, B. J. Taylor, and D. D. Jensen. Causal Discovery in Social Media Using Quasi-experimental Designs. In *Proc. of ACM Workshop on Social Media Analytics*, 2010.
9. J. S. Pudipeddi, L. Akoglu, and H. Tong. User Churn in Focused Question Answering Sites: Characterizations and Prediction. In *Proc. of ACM WWW*, 2014.
10. C. E. Rasmussen. *Evaluation of Gaussian Processes and Other Methods for Non-linear Regression*. PhD thesis, University of Toronto, 1996.
11. C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2005.
12. M. Seeger. Pac-bayesian Generalisation Error Bounds for Gaussian Process Classification. *J. Mach. Learn. Res.*, 3:233–269, 2003.
13. M. Seeger, C. Williams, and N. Lawrence. Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In *Proc. of Workshop on Artificial Intelligence and Statistics*, 2003.
14. E. Snelson and Z. Ghahramani. Sparse Gaussian Processes Using Pseudo-inputs. In *Proc. of NIPS*, 2005.
15. E. Snelson and Z. Ghahramani. Variable Noise and Dimensionality Reduction for Sparse Gaussian Processes. *arXiv preprint arXiv:1206.6873*, 2012.