Decoding Finger Flexion from Electrocorticographic Signals Using a Sparse Gaussian Process

Zuoguan Wang, Qiang Ji
Rensselear Polytechnic Institute
110 Eighth Street
Troy, NY, USA, 12180
{wangz6, jiq}@rpi.edu

Kai J. Miller
Department of Physics
box 351560, University of Washington
Seattle, WA, 98195, USA
kjmiller@gmail.com

Gerwin Schalk
BCI R&D Progr, Wadsworth Ctr, NYS Dept of Health
C650 Empire State Plaza
Albany, NY, 12201, USA
schalk@wadsworth.org

Abstract

A brain-computer interface (BCI) creates a direct communication pathway between the brain and an external device, and can thereby restore function in people with severe motor disabilities. A core component in a BCI system is the decoding algorithm that translates brain signals into action commands of an output device. Most of current decoding algorithms are based on linear models (e.g., derived using linear regression) that may have important shortcomings. The use of nonlinear models (e.g., neural networks) could overcome some of these shortcomings, but has difficulties with high dimensional feature spaces. Here we propose another decoding algorithm that is based on the sparse gaussian process with pseudo-inputs (SPGP). As a nonparametric method, it can model more complex relationships compared to linear methods. As a kernel method, it can readily deal with high dimensional feature space. The evaluations shown in this paper demonstrate that SPGP can decode the flexion of finger movements from electrocorticographic (ECoG) signals more accurately than a previously described algorithm that used a linear model. In addition, by formulating problems in the bayesian probabilistic framework, SPGP can provide estimation of the prediction uncertainty. Furthermore, the trained SPGP offers a very effective way for identifying important features.

1. Introduction

Brain computer interfaces (BCI) decode a user' intent from brain signals [10]. Because a BCI system directly converts brain signals into commands to control machines, it can be used to restore function to people with severe body paralysis. A core component in a BCI system is the decoding algorithm which translates brain signals into action commands to control artificial actuators. Decoding algorithms have received substantial attention in the signal processing and machine learning literature. The basic class of decoding methods is based on linear models of the relationship between brain signals and particular output parameters (e.g., kinematic parameters of limb movements). Of the basic class, the simplest method takes the linear weighted summation of neural activity. A lot of BCI studies have adopted this approach to decode movement parameters from neural activity [4, 8]. Other studies have also used Kalman filters that explicitly characterize the temporal evolution of movement parameters [7, 1]. One important benefit offered by Kalman filter is that as a probabilistic method, kalman filter can provide confidence estimate for results. Other studies have used other versions of linear models such as Pace regression [2] or ridge regression [5]. Due to their relative simplicity and efficacy, these linear methods were commonly used in experimental research in BCI. However, linear models can not handle more complex relationships between brain and output control signals. To better describe these neuronal modulations, a variety of non-linear methods have been developed, which include neural network: [7], multilinear perceptron [3]. But they tend to have difficulty with high dimensional features and limited training data.

A Gaussian process provides a very popular and elegant nonparametric Bayesian model for real world statistical problem. As a kernel based method, it can readily work in high dimensional feature space with kernelized operation in the input space. For it models problems in an explicit probabilistic formulation, Gaussian processes can provide results in parallel with confidence interval (for regression). Unfortunately due to its nonparametric nature, the computational cost of GP increases cubically with the number of training data. However, this drawback has been solved by various approximation methods. Here, we propose to use sparse gaussian process regression using pseudo-inputs for decoding of finger movements using electrocorticographic (ECoG) recorded from the surface of the brain. Compared to results of a recently published study [2], our propopsed algorithm provides three benefits: 1. More accurate decoding was obtained; 2. The uncertainty for prediction was estimated. 3. The contribution of each feature to the regression can be shown from the estimated length scale parameters.

2. Sparse Gaussian Process for Regression

2.1 Gaussian Process for Regression

We provides a brief description of the gaussian process for regression below. (A comprehensive description is given in [6].). Gaussian processes extend finite multivariate Gaussian distribution to infinite space. All the observations in the data set can be seen as a sample from this infinite multivariate Gaussian distribution. Specifically, we have a data set S of N observations, $S = \{(x_n, y_n)|n = 1, \ldots, N\}$, where x denotes an input vector of dimension D and y is its corresponding real valued target. We put a Gaussian process prior on the function $f(x)$ so that given any finite input data set X, $p(f|X)$ follows an multivariate Gaussian distribution with mean m and covariance K. Very often we assume the mean $m = 0$ and the distribution is determined by the covariance K. The covariance K is specified by the covariance function which relates one observation to another and depends on only a small number of parameters. In our experiment, we choose the RBF kernel with ARD hyperparameters:

$$K(x, x') = \sigma_f^2 \exp[-\frac{1}{2}\sum_{d=1}^{D} l_d(x_{(d)} - x'_{(d)})^2] \qquad (1)$$

where l is the length scale parameter and σ_f^2 defines the signal variance. The Gaussian process also assumes that the observation y is corrupted by a gaussian noise, that is:

$$y = f(x) + \mathcal{N}(0, \sigma_n^2) \qquad (2)$$

And by combining the observation noise the kernel function yields:

$$K(x, x') = \sigma_f^2 \exp[-\frac{1}{2}\sum_{d=1}^{D} l_d(x_{(d)} - x'_{(d)})^2] + \sigma_n^2 \delta(x, x'), \qquad (3)$$

where δ is the Kronecker delta function and the hyperparameters $\theta = \{\sigma_f^2, l, \sigma_n^2\}$ are learned from training data by maximizing the likelihood $p(y|X, \theta) = \mathcal{N}(y|0, K)$.

For regression, given input point x_*, we want to infer y_* conditioning on the observed data S and trained parameters θ. According to the Gaussian process prior, the joint distribution of y and y_* is still multivariate Gaussian distribution:

$$\left[\begin{array}{c} y \\ y_* \end{array}\right] \sim \mathcal{N}\left(0, \left[\begin{array}{cc} K & K_*^T \\ K_* & K_{**} \end{array}\right]\right)$$

where $[K_*]_n = K(x_*, x_n)$ and $K_{**} = K(x_*, x_*)$. The prediction on y_* given x_* is obtained by:

$$p(y_*|x_*, S, \theta) = \mathcal{N}(y_*|K_*K^{-1}y, K_{**} - K_*K^{-1}K_*^T). \qquad (4)$$

2.2 Sparse Gaussian Process with Pseudo-Inputs

To better deal with large size data with GP, Snelson and Ghahramani [9] introduced a sparse Gaussian process with pseudo-inputs. The basic idea is that instead of using whole data set $\mathcal{D}$ for training, we only consider a pseudo data set $\bar{\mathcal{D}}$ that consists of $\{\bar{x}_m, \bar{f}_m\}_{m=1}^M$ where $M < N$. The pseudo data set is chosen in the way that it can provide best representation for the original data set. As the standard model discussed in last section, the single data point likelihood under the pseudo data set is:

$$p(y_*|x_*, \bar{X}, \bar{f}) = \mathcal{N}(y_*|K_*K_M^{-1}\bar{f}, K_{**} - K_*K_M^{-1}K_*^T). \qquad (5)$$

where $[K_M]_{mm'} = K(\bar{x}_m, \bar{x}_{m'})$, $[K_*]_m = K(x_*, \bar{x}_m)$ and $K_{**} = K(x_*, x_*)$. The pseudo targets $\bar{f}$ was not explicitly represented, but was integrated out in prediction. So given the new input data x_*, the prediction distribution is given by:

$$\begin{aligned} p(y_*|x_*, \mathcal{D}, \bar{X}) &= \int d\bar{f} p(y_*|x_*, \bar{X}, \bar{f}) p(\bar{f}|\mathcal{D}, \bar{X}) \qquad (6) \\ &= \mathcal{N}(y_*|u_*, \sigma_*^2), \end{aligned}$$

where $u_* = K_*Q_M^{-1}K_{MN}\Lambda^{-1}y$, $\sigma_*^2 = K_{**} - K_*(K_M^{-1} - Q_M^{-1})K_*^T$, $Q_M = K_M + K_{MN}\Lambda^{-1}K_{NM}$, $\Lambda = diag(\lambda)$, $\lambda_n = K_{nn} - K_*K_M^{-1}K_*^T$, $[K_{NM}]_{nm} = K(x_n, \bar{x}_m)$. The problem left is to find the hyperparameters θ and the pseudo-input location $\bar{X}$. This is done by maximizing the marginal likelihood:

$$\begin{aligned} p(y|X, \bar{X}, \theta) &= \int d\bar{f} p(y|X, \bar{X}, \bar{f}) p(\bar{f}|\bar{X}) \qquad (7) \\ &= \mathcal{N}(y|0, K_{NM}K_M^{-1}K_{NM} + \Lambda), \end{aligned}$$

through gradient ascent.

3. Experiments

3.1 Data collection and feature extraction

The following section gives a brief overview of data collection and feature extraction. A more comprehensive description is given in [2]. Signals were collected from 48-64 electrodes placed on the surface of the brain (electrocorticography (ECoG)) in five human subjects

Table 1. Decoding performance comparison between Pace regression and SPGP(mean/min/max).

Subject	Alg.	Thumb	Index Finger	Middle Finger	Ring Finger	Little Finger	Avg.
A	pace	0.70/0.67/0.74	0.68/0.66/0.69	0.61/0.59/0.63	0.56/0.53/0.60	0.59/0.56/0.62	0.63
A	SPGP	0.77/0.74/0.80	0.74/0.67/0.78	0.67/0.60/0.69	0.65/0.58/0.73	0.68/0.61/0.77	0.70
B	pace	0.68/0.63/0.72	0.68/0.52/0.77	0.65/0.55/0.76	0.74/0.70/0.78	0.71/0.58/0.78	0.69
B	SPGP	0.74/0.69/0.77	0.73/0.63/0.80	0.69/0.58/0.79	0.77/0.70/0.81	0.75/0.64/0.81	0.74
C	pace	0.58/0.48/0.65	0.61/0.60/0.62	0.56/0.46/0.60	0.55/0.49/0.60	0.51/0.43/0.58	0.56
C	SPGP	0.58/0.48/0.70	0.68/0.61/0.74	0.65/0.58/0.69	0.62/0.57/0.69	0.59/0.51/0.66	0.62
D	pace	0.35/0.28/0.40	0.46/0.40/0.58	0.50/0.45/0.56	0.45/0.42/0.51	0.38/0.28/0.44	0.42
D	SPGP	0.38/0.29/0.44	0.54/0.47/0.63	0.52/0.44/0.59	0.46/0.40/0.53	0.41/0.38/0.44	0.46
E	pace	0.49/0.46/0.53	0.52/0.42/0.62	0.60/0.51/0/72	0.54/0.47/0.59	0.55/0.47/0.64	0.54
E	SPGP	0.49/0.43/0.54	0.57/0.45/0.64	0.59/0.44/0.69	0.55/0.53/0.60	0.57/0.49/0.65	0.55

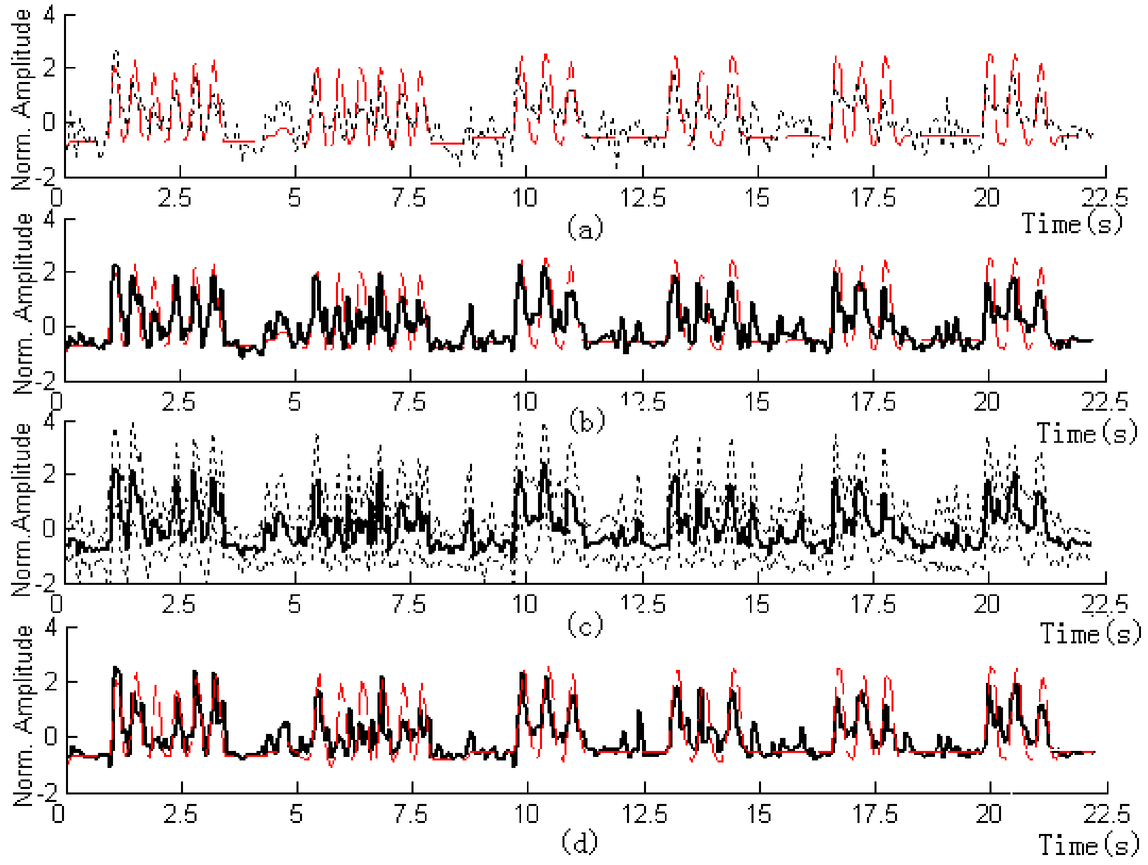

Figure 1. (a) Ground truth (red) and pace regression result (dotted black) with correlation coefficient 0.69; (b) Ground truth (red) and result of SPGP (solid black) with correlation coefficient 0.78; (c) Mean (solid), 2 standard deviation line (dotted black); (d) Prediction with selected 10 features (0.76)

(A-E), while these subjects repetitively flexed a particular finger that was indicated using visual cues. In offline analyses, we first re-referenced the signals to the common average reference (CAR), which subtracted $\frac{1}{H}\sum_{q=1}^{H} s_q$ from each channel, where H was the total number of channels and s_q was the collected signal at the qth channel and at the particular time. For each 100-ms time slice (overlapped by 50 ms) and each channel, we converted the time-series ECoG data into the frequency domain using an autogressive model of order 20. We then used this model to calculate frequency amplitudes between 0 to 1000 Hz in 1 Hz bins. Features were extracted by averaging these frequency amplitudes across specific frequency ranges. In our evaluations, we selected five frequency ranges, that is 8-12Hz, 18-24Hz, 75-115Hz, 125-159Hz, and 159-175Hz. In addition to the frequency features above, a time-domain feature-Local Motor Potential (LMP) was computed by averaging raw time-domain signal at each channel over 100-ms time window, i.e., a total of 6 features. Thus, the total number of features was 288-384 features from the 48-64 channels.

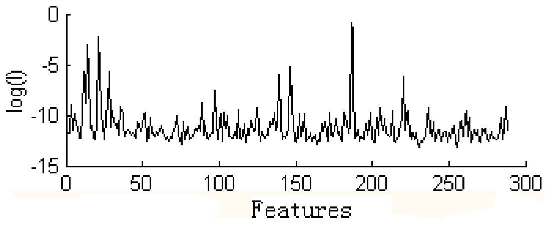

Figure 2. Estimated l for features

3.2 Results

Similar to the procedure described in [2], we defined a movement period as the time between 1000 ms prior to finger movement onset and 1000 ms after movement offset. To compile training data with balanced movement and rest periods, all data outside the movement period were discarded. For each finger, 5-fold cross validation was used to evaluate our model, i.e., 4/5th of data was used for training and 1/5th of data was used for testing. We compared the decoding output with that of Pace regression, i.e., the algorithm that was used previously in [2]. Pace regression was implemented using the Java-based Weka package. Table1 gives the quantitative comparison between the results of SPGP and that of pace regression. It shows that for the large majority of fingers and subjects, SPGP achieved better decoding results. I.e., the overall correlation coefficient calculated between actual and decoded finger flexion improved from 0.56 when using pace regression to 0.61 when using SPGP. Figure 1 gives an example for prediction of pace regression and SPGP on the index finger of subject A. It shows that in the movement part the prediction of SPGP fits the actual flexion better than does the use of pace regression, and the noise in the resting part when using SPGP is lower than when using pace regression. In addition, SPGP can provide uncertainty estimation for the prediction. Figure 1(c) shows the prediction distribution of SPGP with mean and two standard deviations.

Another very important advantage offered by SPGP is that the dimensional scaling parameter l (see eq(1)) provides a very effective way for measuring the importance of features. In fact, l represents the weight of each feature dimension during calculating the covariance. Important features will be assigned with large weights. We can verify the practical utility of these weights by selecting a subset of features corresponding to large weights and comparing the prediction using this subset with that using all the features. When we take the results for the index finger of subject A as an example, figure 2 shows the trained $log(l)$ on the 288 features. Only a few large weights in $log(l)$ make the feature selection very effective. We simply take the features corresponding to the largest 10 weights and rerun the algorithm. The result was shown in figure 1(d) which is almost same with that using the whole 288 features no matter from the graph or from correlation coefficient (0.76 vs 0.78).

4 Conclusion

This paper proposed the use of a sparse gaussian process with pseudo-input for decoding finger flexion from brain signals. As a nonlinear nonparametric bayesian method, it modeled the data better than previously described linear models. In addition, SPGP can provide uncertainty estimation for the prediction. The length scale parameters in the trained SPGP also provide a very effective measurement for the importance of features.

References

[1] M. J. Black, E. Bienenstock, J. P. Donoghue, M. Serruya, W. Wu, and Y. Gao. Connecting brains with machines: The neural control of 2d cursor movement michael j. black. 2003.

[2] J. O. J. W. G. S. J. Kubanek, K.J.Miller. Decoding flexion of individual fingers using electrocorticographic signals in humans. *J Neural Eng*, 6(6):14, 2009.

[3] K. H. Kim, S. S. Kim, and S. J. Kim. Superiority of nonlinear mapping in decoding multiple single-unit neuronal spike trains: A simulation study. *Journal of Neuroscience Methods*, 150(2):202 – 211, 2006.

[4] D. J. McFarland, D. J. Krusienski, W. A. Sarnacki, and J. R. Wolpaw. Emulation of computer mouse control with a noninvasive brain-computer interface. *J Neural Eng*, 5(2):101–110, Mar 2008.

[5] G. H. Mulliken, S. Musallam, and R. A. Andersen. Trajectories from posterior parietal cortex ensembles. *J. Neurosci.*, 28(48):12913–12926, 2008.

[6] C. E. Rasmussen. Gaussian processes for machine learning. MIT Press, 2006.

[7] R. Y. N. P. J. Sanchez J C, Erdogmus D and N. M. A. A comparison between nonlinear mappings and linear state estimation to model the relation from motor cortical neuronal firing to hand movements. pages 59–65, 2002.

[8] G. Schalk and et al. Two-dimensional movement control using electrocorticographic signals in humans. *J Neural Eng*, 5(1):75–84, Mar 2008.

[9] E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *NIPS*, pages 1257–1264. MIT press, 2006.

[10] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain-computer interfaces for communication and control. 113(6):767–791, June 2002.