Deep Kernel Learning

鈴木雅大

論文について

- Wilson et al. (arXiv 2015/11/6)
  - Carnegie Mellon University
  - Salakhutdinovさんも著者の一人
- Deep learning + Gaussian process
- 選んだ理由
  - 現在カーネル&ガウス過程に興味あり
  - タイトルがヤバい

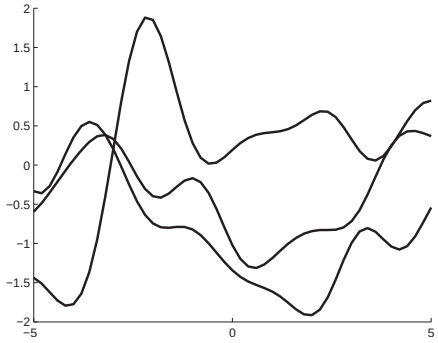
ガウス過程

ガウス過程とは？

- 関数の確率分布 $f(\mathbf{x}) \sim \mathcal{GP}(\mu, k_{\boldsymbol{\gamma}})$
- D次元の入力ベクトルのデータセット$X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$に対する関数の出力ベクトル $\mathbf{y} = (y(\mathbf{x}_1), \ldots, y(\mathbf{x}_n))^\top$ の同時分布が常にガウス分布

$$\mathbf{f} = f(X) = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)]^\top \sim \mathcal{N}(\boldsymbol{\mu}, K_{X,X})$$

  - 平均ベクトルは $\boldsymbol{\mu}_i = \mu(x_i)$ ，共分散行列は $(K_{X,X})_{ij} = k_{\boldsymbol{\gamma}}(\mathbf{x}_i, \mathbf{x}_j)$ で完全に記述される



線形回帰とガウス過程

- 通常の線形回帰は, $\mathbf{y} = \Phi\mathbf{w}$ と書ける.
  - Φは計画行列

$$\underset{\mathbf{y}}{\begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix}} = \underset{\mathbf{\Phi}}{\begin{pmatrix} \phi_1(\mathbf{x}^{(1)}) & \cdots & \phi_H(\mathbf{x}^{(1)}) \\ \phi_1(\mathbf{x}^{(2)}) & \cdots & \phi_H(\mathbf{x}^{(2)}) \\ \vdots & & \vdots \\ \phi_1(\mathbf{x}^{(N)}) & \cdots & \phi_H(\mathbf{x}^{(N)}) \end{pmatrix}} \underset{\mathbf{w}}{\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ \vdots \\ w_H \end{pmatrix}}$$

  - 重みwがガウス分布 $p(\mathbf{w}) = \mathrm{N}(\mathbf{0}, \alpha^{-1}\mathbf{I})$ に従っているとすると，yもガウス分布に従い，平均0，分散 $\alpha^{-1}\mathbf{\Phi}\mathbf{\Phi}^T$ となる.

$$p(\mathbf{y}) = \mathrm{N}(\mathbf{y}|\mathbf{0}, \alpha^{-1}\mathbf{\Phi}\mathbf{\Phi}^T)$$

  - ここで，分散はカーネル関数に置き換えることができる.

$$\mathbf{K} = \alpha^{-1}\mathbf{\Phi}\mathbf{\Phi}^T$$

- つまり，ガウス過程は$f(x)$の事前分布に対応
  - 任意のデータについて成り立つので，ガウス過程は無限次元のガウス分布
  - 重みwは明示的に考えない＝ノンパラメトリック

線形回帰とガウス過程

- 実際には観測値にノイズが入っている.

$$\begin{cases} y = \mathbf{w}^T \phi(\mathbf{x}) + \epsilon \\ \epsilon \sim \mathrm{N}(0, \beta^{-1}\mathbf{I}) \end{cases} \Longrightarrow \quad p(y|f) = \mathrm{N}(\mathbf{w}^T \phi(\mathbf{x}), \beta^{-1}\mathbf{I})$$

- ここで，fについて周辺化する

$$\begin{aligned} p(y|\mathbf{x}) &= \int p(y|f)p(f|\mathbf{x})df \\ &= \mathrm{N}(0, \mathbf{C}) \end{aligned}$$

  - これもガウス分布
  - ただしCは,

$$C(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j) + \beta^{-1}\delta(i, j)$$

- つまり，ガウス過程はカーネル関数とハイパーパラメータで完全に記述できる！

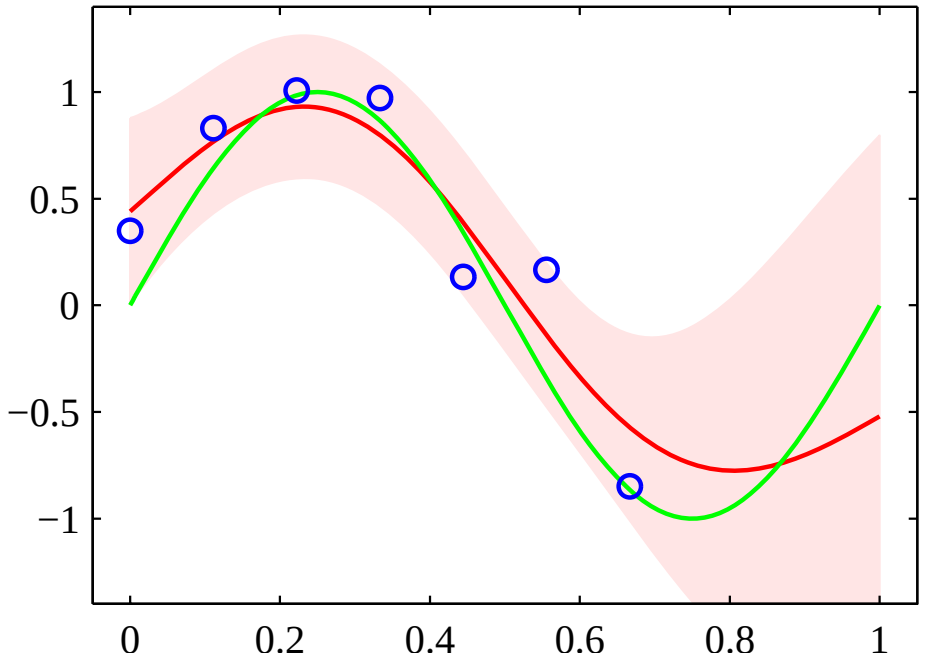
ガウス過程の予測分布

- 新しいy^{new}とこれまでのyの結合分布もガウス分布.
- よって，予測分布は

$$
\begin{aligned}
& p(y^{\mathrm{new}}|\mathbf{x}^{\mathrm{new}}, \mathbf{X}, \mathbf{y}, \theta) \\
&= \frac{p((\mathbf{y}, y^{\mathrm{new}})|(\mathbf{X}, \mathbf{x}^{\mathrm{new}}), \theta)}{p(\mathbf{y}|\mathbf{X}, \theta)} \\
&\propto \exp\left(-\frac{1}{2}([\mathbf{y}, y^{\mathrm{new}}] \begin{bmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^T & k \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{y} \\ y^{\mathrm{new}} \end{bmatrix} - \mathbf{y}^T\mathbf{K}^{-1}\mathbf{y})\right) \\
&\sim \mathrm{N}(\mathbf{k}^T\mathbf{K}^{-1}\mathbf{y}, k - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k})
\end{aligned}
$$

  - ただし　$\mathbf{K} = [k(\mathbf{x}, \mathbf{x}')]$.
    $\mathbf{k} = (k(\mathbf{x}^{\mathrm{new}}, \mathbf{x}_1), \cdots, k(\mathbf{x}^{\mathrm{new}}, \mathbf{x}_N))$
  - カーネル関数は任意に設定，パラメータは経験ベイズで学習する.

予測分布



- モデルを仮定せずにデータから自動的に学習
  - データの少ないところが分散が大きくなっている=曖昧さ

ガウス過程のメリット&デメリット

- メリット
  - 任意のカーネルを設定できる
  - 無限次元の特徴量を考えることができる（ガウスカーネル）
  - 予測分布について，データ点ごとの分散を考えることができる

- デメリット
  - 訓練データをすべて保持する必要がある（$O(n^2)$）．
  - 計算量が多い
    - 学習： $O(n^3)$
    - 予測：平均$O(n)$，分散$O(n^2)$

ガウス過程とニューラルネットワーク

- ガウス過程のノンパラメトリックな特徴をニューラルネットワークで！
  - “How can Gaussian processes possibly replace neural networks? Have we thrown the baby out with the bathwater?” [MacKay1998]
  - めんどいパラメータを自動調節したい（ちょうどみんなイライラしてた頃）.
- Bayesian Learning for Neural Networks [Neal 1996]
  - NealのD論（指導教官はHinton，実質的相談相手はMacKey）
  - 機械学習のコミュニティでガウス過程が利用された初めての例
  - ニューラルネットワークの有限な基底関数を無限の基底関数に置き換えた.
- 近年では，表現力の高いカーネル関数が提案されている
  - [Wilson, 2014][Wilson and Adams, 2013][Lloyd et al., 2014][Yang et al., 2015]
  - 人間の介入なしで，データから高次元の構造を学習できる.
  - ただし，これはニューラルネットワークを置き換えるということではない
  - むしろ組み合わせることが大事！（本研究の目標）

関連研究

ガウス過程+ニューラルネットワークはこれまでも様々提案されている.

- Gaussian process regression network [Wilson et al., 2012]
  - 全ての重みをガウス過程で置き換えた
- Damianou and Lawrence (2013)
  - 全ての活性化関数をガウス過程に.
- Salakhutdinov and Hinton (2008)
  - DBNとガウス過程の融合.
- Calandra et al. (2014)
  - NNとガウス過程の融合， sharp discontinuitiesを学習可能

ただし，どの手法も数千以上訓練データを大きくできない.

提案手法

- kを基底カーネルとして，入力xからの変換を次のように表す．

$$k(\mathbf{x}_i, \mathbf{x}_j|\boldsymbol{\theta}) \rightarrow k(g(\mathbf{x}_i, \mathbf{w}), g(\mathbf{x}_j, \mathbf{w})|\boldsymbol{\theta}, \mathbf{w})$$

  - ただしgを非線形写像（DNN）とする（wは重みなどのパラメータ）．

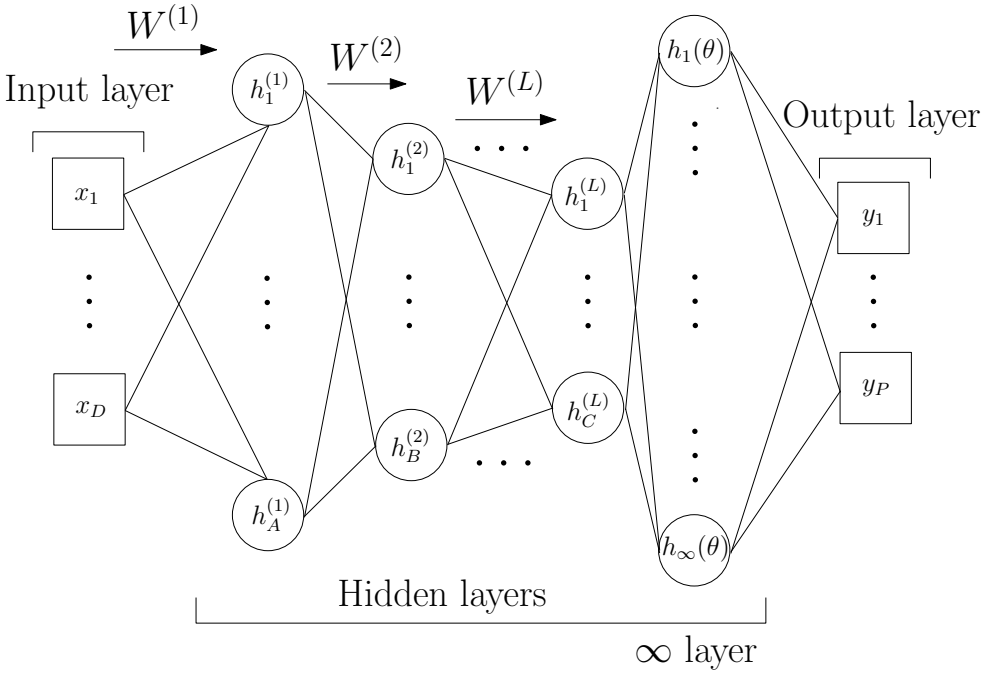
- 本研究ではRBFカーネルの他，spectral mixture (SM) カーネル [Wilson and Adams, 2013]を利用

$$k_{\mathrm{SM}}(\mathbf{x}, \mathbf{x}'|\boldsymbol{\theta}) = \sum_{q=1}^{Q} a_q \frac{|\Sigma_q|^{\frac{1}{2}}}{(2\pi)^{\frac{D}{2}}} \exp\left(-\frac{1}{2}||\Sigma_q^{\frac{1}{2}}(\mathbf{x}-\mathbf{x}')||^2\right) \cos\langle \mathbf{x}-\mathbf{x}', 2\pi\boldsymbol{\mu}_q\rangle$$

  - 準周期定常を見つけることができる．
  - 一方DNNの写像gでは非定常で階層的な構造を捉える

提案手法

- 先ほどのディープなカーネルをガウス過程の共分散行列とする
  - 最終層がガウス過程になったDNNと解釈できる
  - カーネルをRBFやSMにした場合無限の基底関数となるので，無限の隠れユニットを考えることと同じになる．



提案手法の学習

- ガウス過程の周辺尤度Lを最大化する（経験ベイズ）

$$\log p(\mathbf{y}|\boldsymbol{\gamma}, X) \propto -[\mathbf{y}^\top (K_{\boldsymbol{\gamma}} + \sigma^2 I)^{-1}\mathbf{y} + \log |K_{\boldsymbol{\gamma}} + \sigma^2 I|]$$

- カーネルのパラメータθとDNNのパラメータwについて偏微分

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{L}}{\partial K_{\boldsymbol{\gamma}}} \frac{\partial K_{\boldsymbol{\gamma}}}{\partial \boldsymbol{\theta}}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}}{\partial K_{\boldsymbol{\gamma}}} \frac{\partial K_{\boldsymbol{\gamma}}}{\partial g(\mathbf{x}, \mathbf{w})} \frac{\partial g(\mathbf{x}, \mathbf{w})}{\partial \mathbf{w}}$$

  - カーネルによる偏微分の部分は

$$\frac{\partial \mathcal{L}}{\partial K_{\boldsymbol{\gamma}}} = \frac{1}{2}(K_{\boldsymbol{\gamma}}^{-1}\mathbf{y}\mathbf{y}^\top K_{\boldsymbol{\gamma}}^{-1} - K_{\boldsymbol{\gamma}}^{-1})$$

KISS-GP共分散行列

- カーネルKをKISS-GP共分散行列で近似 [Wilson and Nickisch, 2015] [Wilson et al., 2015]

$$K_{\boldsymbol{\gamma}} \approx M K_{U,U}^{\mathrm{deep}} M^{\top} := K_{\mathrm{KISS}}$$

  - M:補完重みのスパース行列
  - K:ディープカーネルから求める
  - 予測$K_{\mathrm{KISS}}^{-1}\mathbf{y}$の際は，linear conjugate gradients (LCG) を使う
  - その他細い話は省きます.

- 他の近似手法に比べて計算量が少ない&高性能
  - $\mathcal{O}(n+h(m))$
    - 変分近似の手法[Quiñonero-Candela and Rasmussen, 2005] だと $\mathcal{O}(m^2 n + m^3)$ かつ精度も良くない

実験

- 本研究では次の3つのタスクで実験
  - UCI repositoryの回帰問題
  - 顔画像の回転角度抽出
  - MNIST画像の回帰問題
  - ステップ関数の回帰問題

- DNNはCaffe，KISS-GPはGPML で実装

- DNNの学習はSGD，活性化関数はReLU
  - DNNを事前学習した後，出力をKISS-GPの入力とする．
  - 学習は周辺尤度を偏微分して更新

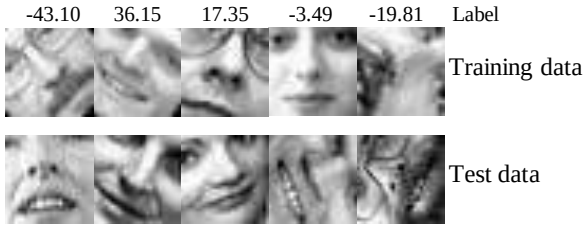
UCI repositoryの回帰問題

- ネットワーク構造
  - n<6000：[d-1000-500-50-2]
  - n>6000：[d-1000-1000-500-50-2]
- 厳密なGPはデータセットが膨大な時は扱えないので，Fastfood finite basis function expansionsで推定.

Datasets	n	d	RMSE						Runtime(s)		
			GP			DNN	DKL		DNN	DKL	
			RBF	SM	best		RBF	SM		RBF	SM
Gas	2,565	128	0.21±0.07	0.14±0.08	0.12±0.07	0.11±0.05	0.11±0.05	**0.09±0.06**	7.43	7.80	10.52
Skillcraft	3,338	19	1.26±3.14	0.25±0.02	0.25±0.02	**0.25±0.00**	**0.25±0.00**	**0.25±0.00**	15.79	15.91	17.08
SML	4,137	26	6.94±0.51	0.27±0.03	0.26±0.04	0.25±0.02	0.24±0.01	**0.23±0.01**	1.09	1.48	1.92
Parkinsons	5,875	20	3.94±1.31	**0.00±0.00**	**0.00±0.00**	0.31±0.04	0.29±0.04	0.29±0.04	3.21	3.44	6.49
Pumadyn	8,192	32	1.00±0.00	0.21±0.00	**0.20±0.00**	0.25±0.02	0.24±0.02	0.23±0.02	7.50	7.88	9.77
PoleTele	15,000	26	12.6±0.3	5.40±0.3	4.30±0.2	3.42±0.05	3.28±0.04	**3.11±0.07**	8.02	8.27	26.95
Elevators	16,599	18	0.12±0.00	0.090±0.001	0.089±0.002	0.099±0.001	**0.084±0.002**	**0.084±0.002**	8.91	9.16	11.77
Kin40k	40,000	8	0.34±0.01	0.19±0.02	0.06±0.00	0.11±0.01	0.05±0.00	**0.03±0.01**	19.82	20.73	24.99
Protein	45,730	9	1.64±1.66	0.50±0.02	0.47±0.01	0.49±0.01	0.46±0.01	**0.43±0.01**	142.8	154.8	144.2
KEGG	48,827	22	0.33±0.17	0.12±0.01	0.12±0.01	0.12±0.01	0.11±0.00	**0.10±0.01**	31.31	34.23	61.01
CTslice	53,500	385	7.13±0.11	2.21±0.06	0.59±0.07	0.41±0.06	0.36±0.01	**0.34±0.02**	36.38	44.28	80.44
KEGGU	63,608	27	0.29±0.12	0.12±0.00	0.12±0.00	0.12±0.00	**0.11±0.00**	**0.11±0.00**	39.54	42.97	41.05
3Droad	434,874	3	12.86±0.09	10.34±0.19	9.90±0.10	7.36±0.07	**6.91±0.04**	**6.91±0.04**	238.7	256.1	292.2
Song	515,345	90	0.55±0.00	0.46±0.00	0.45±0.00	0.45±0.02	0.44±0.00	**0.43±0.01**	517.7	538.5	589.8
Buzz	583,250	77	0.88±0.01	0.51±0.01	0.51±0.01	0.49±0.00	0.48±0.00	**0.46±0.01**	486.4	523.3	769.7
Electric	2,049,280	11	0.230±0.000	0.053±0.000	0.053±0.000	0.058±0.002	0.050±0.002	**0.048±0.002**	3458	3542	4881

顔画像の回転角度抽出

- 顔の回転角度のラベルを予測する
  - The Olivetti face data をランダムに回転，クロップ，サブサンプリングして28×28のデータとしたもの[Salakhutdinov and Hinton (2008)]を利用
  - 30人で学習，10人を予測



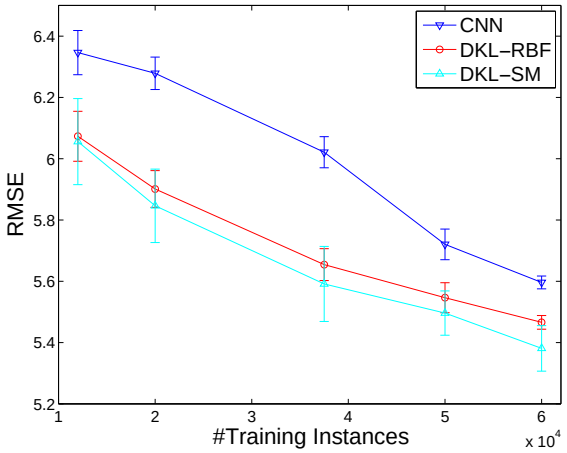
- ネットワーク構造
  - 2層の畳み込み層，4層の全結合層
  - 出力は2次元

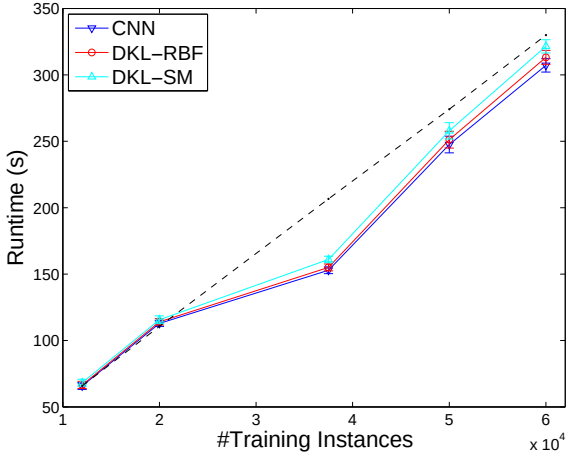
Layer	conv1	pool1	conv2	pool2	full3	full4	full5	full6
kernel size	5×5	2×2	5×5	2×2	-	-	-	-
stride	1	2	1	2	-	-	-	-
channel	20	20	50	50	1000	500	50	2

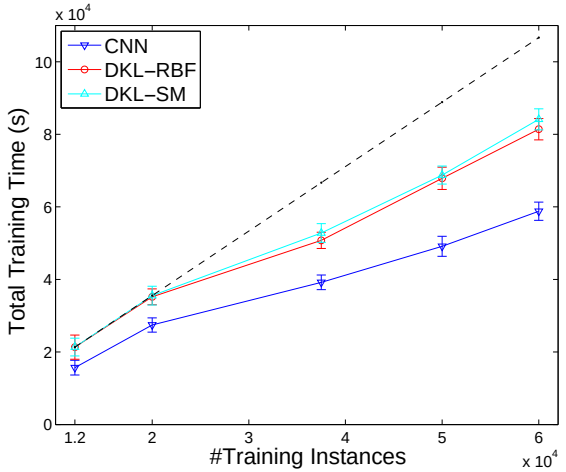
顔画像の回転角度抽出

- 実験結果
  - DBN-GPでは訓練データ12000のうち1000しかラベルを用いていない（残りは教師なし）．一方DKLでは全部のラベルを利用.
  - 平均二乗誤差(RMSE)での評価

Datasets	GP	DBN-GP	CNN	DKL
Olivetti	16.33	6.42	6.34	**6.07**

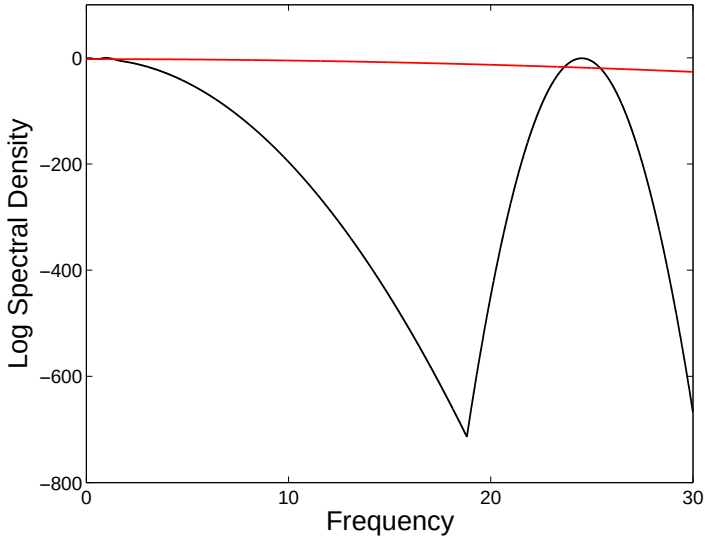






顔画像の回転角度抽出

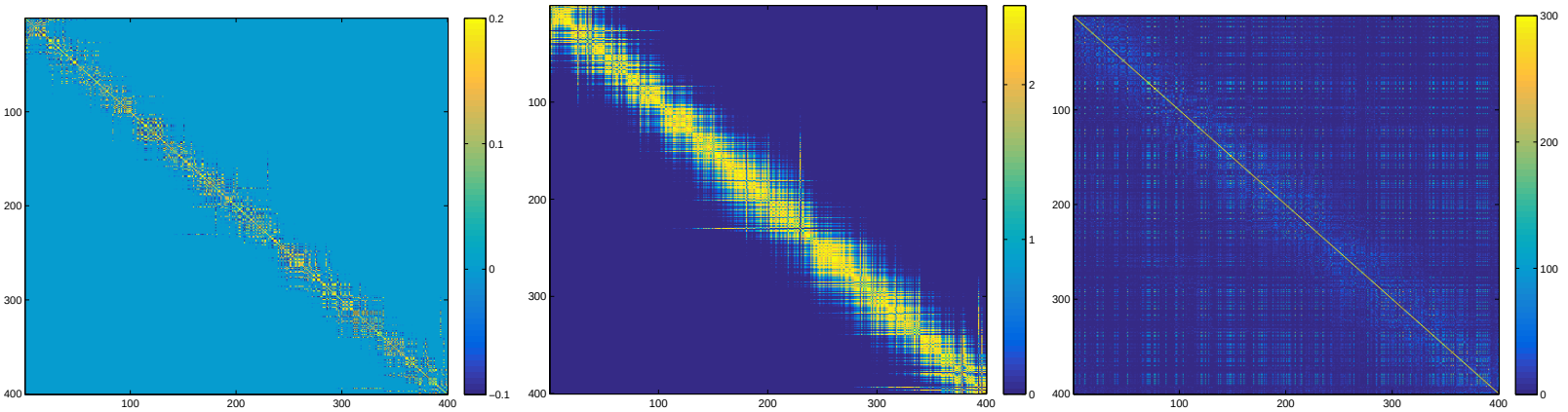
- 基底関数のspectral密度（フーリエ変換）の確認
  - 赤：spectral mixture，黒：RBF



- SMでは，2つのピークを捉えられている．
- 一方RBFではこの重要な局所的相関関係を捉えらえていない．
  - 間違って多くの特徴を捨ててしまう．

顔画像の回転角度抽出

- カーネル共分散行列（テストデータを回転順に並べたもの）
  - 左：DKL-SMカーネル
  - 中央：DKL-RBFカーネル
  - 右：RBFカーネル



- DKLは相関がある
  - DKL-RBFは多少拡散している
- RBFはかなり拡散している

→DKLによって，通常のカーネルよりも相関関係をうまく学習できた

MNIST画像の回帰問題

- MNIST画像で表されている数字にできるだけ近い実数への写像を学習
- ネットワーク構造

Layer	conv1	pool1	conv2	pool2	full3	full4	full5	full6
kernel size	5×5	2×2	5×5	2×2	-	-	-	-
stride	1	2	1	2	-	-	-	-
channel	20	20	50	50	1000	500	50	2

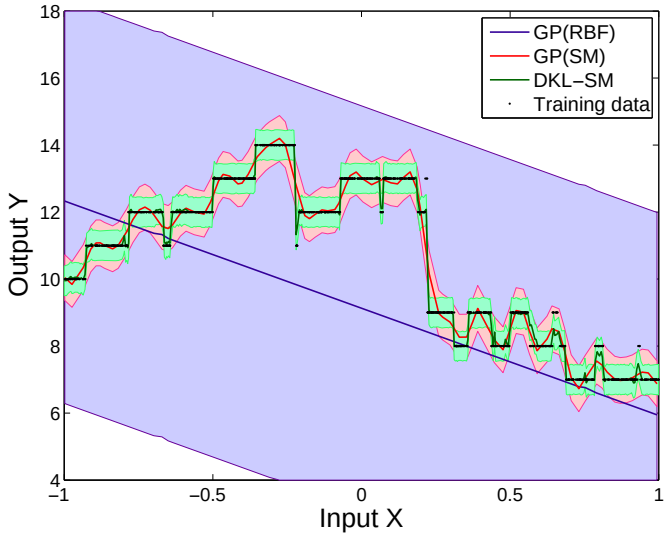
- 実験結果

Datasets	GP	DBN-GP	CNN	DKL
MNIST	1.25	1.03	0.59	**0.53**

- 平均二乗誤差(RMSE)での評価
- 他の手法に比べて良い精度

ステップ関数の回帰

- DKLは通常のDeep Learningと違い事後予測分布を生成できる.
  - 強化学習やベイズ最適化に利用可能
- この問題設定ではステップ関数の回帰を学習する
  - 不連続なので難しい.
  - 通常のカーネルでは滑らかであることを前提としている.
- 実験結果



まとめ

- 本研究ではスケーラブルなディープカーネルを調査
  - 構造的な特性のあるDLとノンパラメトリックな柔軟性を融合
- 基底カーネルの入力をDLに置き換えて，KILL-GPを利用
  - これにより計算量が少なくなった.
- Spectral mixtureカーネルを利用
  - 表現力がさらに高まった.
- 通常のカーネルでもDKLによって表現力が高まり，効率が上がる.
- 様々な実験によって通常のGPやDNNよりも精度が高くなることを示した.