PV Score Algorithm and Dropout for Super-level Set Estimation

2016011165 无 67 江振宇

2019 年 2 月 4 日

目录

1 摘要

本文针对 Super-level Set Estimation 问题，总结了已有的一些方法，提出了加速计算过程的 dropout 方法和新的观测点选择指标 PV score ，并进行了数值仿真实验进行比较。

2 介绍

2.1 问题阐述

很多工程和科学问题涉及到寻找一个未知函数在定义域 Ω 上的最大值。然而，有些应用需要寻找 Ω 的一个尽量大的子空间，满足函数在这个子空间上大于一个阈值 [5]。这个问题可以描述为如下的数学形式:

$$\{\mathbf{x} \in \Omega | P(\mathbf{x} > t) > \delta\}$$

表示找到函数定义域的一个子集，在该子集上，函数值大于阈值的概率大于 δ 。该子集就是 Super-level Set (上水平集)。在这个过程中，需要对这个未知函数在不同自变量上的值进行观测以获得这个函数的信息，而在观测时，我们只能观测到噪声干扰后的函数值。因此我们需要对这个未知函数建立一个概率模型，在本文中，我们涉及到的都是将未知函数建立为高斯过程模型。

2.2 问题定义

为了方便后续的研究，将本问题以如下方式进行定义。

存在一个未知函数 $f(\mathbf{x})$ ，是 $\mathbb{R}^n \to \mathbb{R}$ 的映射。对于这个函数，我们只能通过观测的方式来建立认识。在观测定义域内某点 $\mathbf{x}$ 的函数值时，我们观测的结果是 $f(\mathbf{x}) + \epsilon$ ，其中 $\epsilon \sim N(0, \sigma_\epsilon)$ ，是与观测位置独立的噪声。

2.3 高斯过程

将未知函数建模为高斯过程,每个点的函数值的先验分布服从联合高斯分布,记为 $\mathcal{GP}(\mu_0(\mathbf{x}), k_0(\mathbf{x}, \mathbf{x}'))$。其中 $\mu_0(\mathbf{x})$ 为均值，$k_0(\mathbf{x}, \mathbf{x}')$ 为相关函数（核函数）[3]。

获得 n 个观测值序列 $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ （其中 $y_i = f(\mathbf{x}_i) + \epsilon$）后，高斯过程模型的后验分布为:

$$\begin{aligned} \mu_n(\mathbf{x}) &= \mu_0(\mathbf{x}) + k_n(\mathbf{x})^T (K_n + \sigma_\epsilon^2 I)^{-1} (y_{1:n} - \mu_0(\mathbf{x}_{1:n})) \\ k_n(\mathbf{x}, \mathbf{x}') &= k_0(\mathbf{x}, \mathbf{x}') - k_n(\mathbf{x})^T (K_n + \sigma_\epsilon^2 I)^{-1} k_n(\mathbf{x}') \end{aligned} \tag{1}$$

其中，$k_n = [k_0(\mathbf{x}, \mathbf{x}_1), \cdots, k_0(\mathbf{x}, \mathbf{x}_n)]^T$ ，$K_n = [k_0(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^n$ ，在 $\mathbf{x}$ 处后验方差为 $\sigma_n^2(\mathbf{x}) = k_n(\mathbf{x}, \mathbf{x})$ 。

后验分布也是高斯分布，如此，通过不断的观测，可以逐渐建立对于未知函数的认识。

2.4 上水平集估计

从上面的分析可以看出，高斯过程的参数完全由之前选择的观测点和观测值决定。根据这些参数，可以获得定义域每个点的函数值对应的边缘分布（一维高斯分布）的参数，后验均值 μ 和后验方差 σ^2。根据高斯分布的性质，想要随机变量以 δ 的置信度大于阈值 t ，只需满足下述条件即可：

$$\mu - \beta\sigma > t$$

即可，也就是说，满足上式的定义域中的点构成上水平集。式子中 β 与 δ 满足下述关系：

$$\delta = \int_{-\beta\sigma}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

后面所有实验中，都将这里的 β 取为 1.96，对应 97.5% 的置信度。

在进行实验时，因为函数定义域一般是连续的，所以将定义域划分成网格，取所有的格点集合作为新的定义域。衡量模型的准确率时，根据后验分布的均值与阈值的相对大小将点分入上水平集（下水平集），计算下列指标，最后根据 F1-score 评判准确率。

$$\begin{aligned} TP &= |\{x \in \Omega | \mu(x) > t, f(x) > t\}| \\ FP &= |\{x \in \Omega | \mu(x) > t, f(x) \leq t\}| \\ FN &= |\{x \in \Omega | \mu(x) \leq t, f(x) > t\}| \\ \text{F1-score} &= \frac{2TP}{2TP + FP + FN} \end{aligned} \tag{2}$$

2.5 Active Learning

在现实问题中，常常会出现这样的情形：对每一个点的观测消耗很大。因此，如何在有限的观测数内获得尽可能准确，尽可能大的上水平集，成为一个值得探索的问题。可以通过一个点一个点地观测，根据已有的观测获得的模型参数，去寻找下一个可能收益最高的点，再去观测这一个点的函数值，以此类推。这个过程就是一个 Active Learning 得过程：模型主动选择下一个训练数据，得到其 Label 。

至此，可以得到利用 Active Learning 的思想和高斯过程的模型对未知函数进行估计的算法框架：

Algorithm 1 Active Learning

1: **Input**: 初始均值 μ_0, 初始相关函数k_0, 要观测的函数f, 阈值t, 置信度δ
2: *loop*:
3: $\mathbf{x}^+ \leftarrow SelectPoint(GP)$
4: 观测 $\mathbf{x}^+$ 点函数值获得 y^+
5: 用新观测的 $\mathbf{x}^+, y^+$ 更新高斯过程，$GP^+ \leftarrow GP$
6: **Output**: 估计的上水平集 I_{GP}

3 已有工作

由上面的分析可以看出，算法的核心就是如何选择下一个观测点，在选择下个观测点时，常常会面临利用 (exploitation) 和探索 (exploration) 的抉择困难，利用就是在已经观测的区域附近，利用已有的信息，进一步观测；而探索则是在离已观测的区域较远的区域进行观测。在贝叶斯优化问题（寻找最值）中，这一困境尤为显著，利用可能会导致趋于局部极值，而探索可能会导致浪费观测机会。

下面简单探讨一下已有的相关工作。

3.1 straddle heuristic

straddle heuristic 是一种启发式算法 [1]，定义了 straddle score:

$$straddle(\mathbf{x}) = 1.96\sigma_{n-1}(\mathbf{x}) - |\mu_{n-1}(\mathbf{x}) - t|$$

在选择下一个点时，选取 straddle score 最高的点，也就是说：

$$x^+ = \underset{x\in\Omega}{\operatorname{argmax}}\, straddle(\mathbf{x})$$

可以观察到，straddle score 和方差正相关，和均值与阈值的距离负相关，相关函数值一般与两个点的距离负相关，因此离之前观测过的点越远，条件方差越大，不确定度越大。因此，straddle heuristic 会优先选择当前训练出的模型中，离阈值近的点中，与已观测点较远的点。

可以看出，straddle heuristic 算法更趋向于探索，也导致定义域边界附近的点被观测的几率更大，因为越靠近边界越可能离已有的观测点远。

3.2 LSE

LSE (Level Set Estimation) 算法本质上与 straddle heuristic 算法相似 [2]，也在试图寻找离阈值最近，最模棱两可的点作为下一个观测点。不同之处在于，LSE 算法不会从已经可以划分到上水平集或者下水平集里的点中选取观测点。

LSE 算法中定义了置信区间 $Q_n(\mathbf{x}) = [\mu_{n-1}(\mathbf{x}) - \beta\sigma_{n-1}(\mathbf{x}), \mu_{n-1}(\mathbf{x}) + \beta\sigma_{n-1}(\mathbf{x})]$ ，并定义置信区域 $C_n(\mathbf{x}) = \cap_{i=1}^{n} Q_i(\mathbf{x})$。当某点置信区域位于 $t-\epsilon$ 之上或者 $t+\epsilon$ 之下时，该点就可以被划分到上水平集（下水平集），这里的 ϵ 是一个技术性参数，用于提高算法表现。

LSE 算法选择下一个观测点的指标与 straddle heuristic 相同。

3.3 RMILE

RMILE(Robust Maximum Improvement for Level-set Estimation) 与上述启发式的工作不同，是从解析的角度推导如何才能得到最大的上水平集。此外，考虑到模型可能陷入利用困境，考虑一个极限情况，先验分布将所有的点划分到上水平集，那么模型为了保持最大的上水平集，会不断地

去观测模型中均值最大的点，因此在指标中加入激励探索的项。最终，下一个观测点的选择遵从如下规律：

$$\underset{\mathbf{x}^+\in\Omega}{\operatorname{argmax}} E_{GP}(\mathbf{x}^+)$$
$$E_{GP}(\mathbf{x}^+) = \max\{E_{y^+}|I_{GP^+}| - |I^{\epsilon}_{GP}|, \gamma\sigma_{GP}(\mathbf{x}^+\} \tag{3}$$

其中，I_{GP} 表示由 GP 模型所推断的上水平集，I^{ϵ}_{GP} 表示阈值为 $t-\epsilon$ 时表示由 GP 模型所推断的上水平集。

根据高斯过程的后验分布公式1可以知道，后验方差完全由之前观测点的坐标决定，与观测值无关。因此可以在进行观测之前，就可以根据高斯过程模型得到任一点当前的均值，方差和观测新点之后的方差，所以可以对观测 x^+ 后的上水平集进行计算，结果如下：

$$E_{y^+}(I_{GP^+}) = \sum_{x\in\Omega}\Phi(\frac{\sqrt{\sigma^2_{GP}(x^+)+\sigma^2_{\epsilon}}}{|Cov_{GP}(f(x),f(x^+))|}\times(\mu_{GP}(x)-\beta\sigma_{GP^+}-t))$$

再根据前述规律遍历定义域计算分数，即可找到选出下一个观测点。

4 方法

4.1 PV score

下面引入一种新的观测点选择指标，PV score 。这里 P 指的是误判概率 p_{err}，而 V 指的是方差。这个指标是误判概率与标准差的加权和。其中，误判概率定义如下，对一个定义域中的点，根据当前模型，计算均值和方差，如果均值大于阈值，误判率为随机变量值小于阈值的概率，否则为随机变量值大于阈值的概率。

选择误判概率作为指标是非常直观的，因为我们的最终衡量标准是准确率。在选取坐标点时，我们所拥有的唯一信息就是由之前的观测值所唯一决定的高斯过程模型，要提高准确率，在该模型所描述的分布下选择误判概率最高的点观测是非常自然的想法。

然而仅凭误判概率会导致选择时缺少探索的动力，选择未探索过的区域的点的概率较小，因此加入方差项。方差仅仅与观测位置有关，与观测值无关。考虑二维联合高斯分布的条件方差：

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma^{-1}_{22}\Sigma_{21}$$

可以看到，在相关函数绝对值随两点距离增加而减小的条件下，离已观测点越远的点条件方差越大，这点也可以推广到高维分布。所以加入方差项，可以提高下一个点选在为观测过的区域的可能性。

PV score 具体表达式如下：

$$pv_n(\mathbf{x}) = p_{err}(\mathbf{x}) + \beta\sigma_{n-1}(\mathbf{x})$$

$$\mathbf{x}^+ = \underset{\mathbf{x}\in\Omega}{\operatorname{argmax}}\, pv_n(\mathbf{x})$$

其中，β 是加权参数，而误判概率指的是，对一个定义域中的点，根据当前模型，计算均值和方差，如果均值大于阈值，误判率为随机变量值小于阈值的概率，否则为随机变量值大于阈值的概率，数学表达式如下：

$$\begin{aligned} p_{err}(\mathbf{x}) &= \int_{|\mu(\mathbf{x})-t|}^{\infty} \Phi(\mathbf{x}) \\ &= \int_{|\mu(\mathbf{x})-t|}^{\infty} \frac{1}{\sqrt{2\pi}\sigma(\mathbf{x})} e^{-\frac{x^2}{2\sigma(\mathbf{x})^2}} dx \\ &= \int_{|\mu(\mathbf{x})-t|/\sigma(\mathbf{x})}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt \\ &= 1 - \int_{-\infty}^{|\mu(\mathbf{x})-t|/\sigma(\mathbf{x})} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt \end{aligned} \tag{4}$$

由上面的推导可以看到，误判概率就是 1 减去标准高斯分布的累计分布函数在 $\frac{|\mu(\mathbf{x})-t|}{\sigma(\mathbf{x})}$ 处的函数值，可通过查表计算，复杂度不大。

除此之外，本算法也引入 LSE 中的置信区间来避免重复采样，具体算法如下：

Algorithm 2 PV score

Input: 定义域 D ，先验高斯分布 GP (μ_0, k, σ_0) ，阈值 h ，误差参数 ϵ
$H_0 \leftarrow \emptyset, U_0 \leftarrow D$
$t \leftarrow 1$
loop:
  $H_t \leftarrow H_{t-1}, U_t \leftarrow U_{t-1}$
  for all $x \in U_{t-1}$ **do**
    if $\min(Q_t(x)) + \epsilon > h$ **then**
    $U_t \leftarrow U_t \setminus \{x\}, H_t \leftarrow H_t \bigcup \{x\}$
    else if $\max(Q_t(x)) - \epsilon < h$ **then**
    $U_t \leftarrow U_t \setminus \{x\}$
  $x_t = (\operatorname{argmax})_{x\in U_t} 1.96\sigma_{t-1}(\mathbf{x}) - |\mu_{t-1}(\mathbf{x}) - t|$
  观测 x_t 点函数值获得 y_t
  用新观测的 x_t, y_t 更新高斯过程，$GP_t \leftarrow GP_{t-1}$

4.2 dropout

在前面提到的算法中，在选择下一个观测点时，一般需要遍历定义域中的所有点，计算 $k_n = [k_0(\mathbf{x}, \mathbf{x}_1), \cdots, k_0(\mathbf{x}, \mathbf{x}_n)]^T$，再由此计算方差和均值，进而根据指标选择下一个观测点，当已观测点越来越多时，计算复杂度会越来越高。事实上，由于未知函数和相关函数本身是连续的，而观测本身是带噪声的，因此对定义域中每个点都计算有些冗余。受到到神经网络中 dropout 层的启发 [4]，

我在计算中也引入 dropout 的思想。即在遍历定义域的点时，以一定概率 p 跳过该点，不进行计算。如此，平均来看，只需要计算 $(1-p)|\Omega|$ 个点的相关数值即可，减小了运算量。

可见，引入 dropout 确实减小了计算复杂度，下面定性分析 dropout 对于模型表现的影响。下面我们考虑定义域是二维的情形。将指标函数记为 $score(p)$ ，选取指标最大的点作为下一个观测点。设 p_{max} 为不引入 dropout 时，应当选择的观测点。考虑到模型的连续性，以及现实使用中的离散选点模型，可以假设与 p_{max} 相邻的 8 个点的指标函数值与 $score(p_{max})$ 的差值小于其他点的函数值与 $f(p_{max})$ 的差值，也就是说，这 9 个点的指标函数值比定义域中其他点都要大。因此，引入 dropout 之后，选择这 9 个点以外的点作为下一个观测点的概率等于这 9 个点全部被跳过的概率 p^9 。在 $p=0.5$ 时，选择这 9 个点（p_{max} 邻域中的点）作为下一个观测点的概率是 0.998 ，而选择 p_{max} 邻域中的点作为观测点，对于模型的影响是相当小的。由此可见，引入 dropout 并不会明显影响模型表现，后续实验会进一步说明。

5 实验

下面进行上述几种方法的对比实验，比较 straddle heuristic, LSE 和 PV score 算法的表现，β 参数的选择对于 PV score 算法表现的影响以及 dropout 参数对于运算时间和效果的影响。

5.1 实验设定

实验中均采用 sinusoidal function 作为未知函数，定义域为 $[0,1]\times[0,2]$ ，表达式为 $f(\mathbf{x})=\sin(10x_1)+\cos(4x_2)-\cos(3x_1x_2)$ ，阈值设为 $t=1$，观测噪声的标准差为 $\sigma_\epsilon=e^{-1}$ 。核函数为 $k(\mathbf{x},\mathbf{x}')=\sigma_{ker}^2 e^{-\frac{|\mathbf{x}-\mathbf{x}'|^2}{2l^2}}$ ，是距离的函数，恒正，单调减，其中，$\sigma_{ker}=e, l=e^{-1.5}$。将连续的定义域平均划分为 31 * 61 个格点作为离散的定义域。每个实验重复运行 5 次。

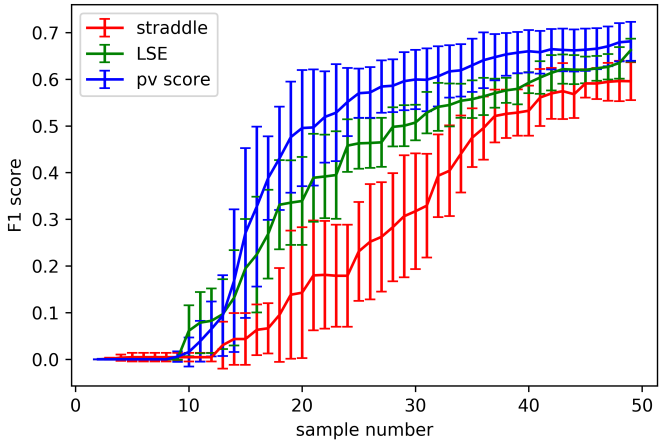

图 1: F1 score 比较

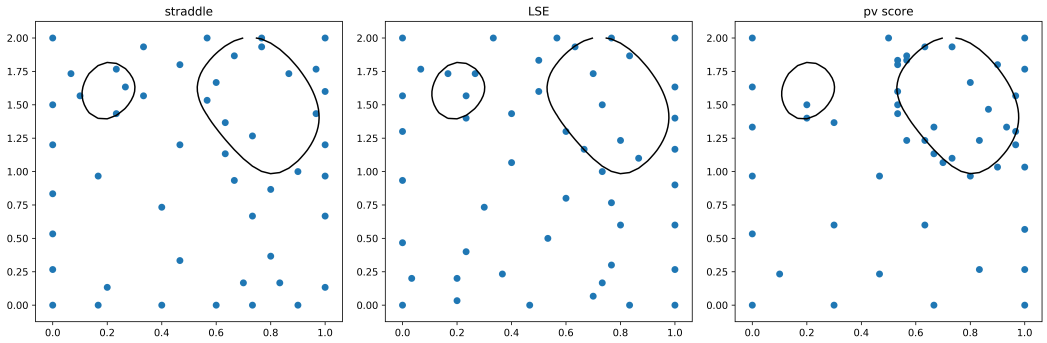

图 2: 观测点位置比较

5.2 算法对比

LSE 和 PV score 算法中，$\epsilon = 10^{-12}$，PV score 中 $\beta = 0.2$，实验结果如图1，可以看到 PV score 算法整体上略优于 LSE 和 straddle heuristic 算法。而三种算法，选择的观测点的分布如图2，可以看到，PV score 会更多地选择在阈值边界上的点进行观测，区分这些模棱两可的点能获得比较大的信息收益。

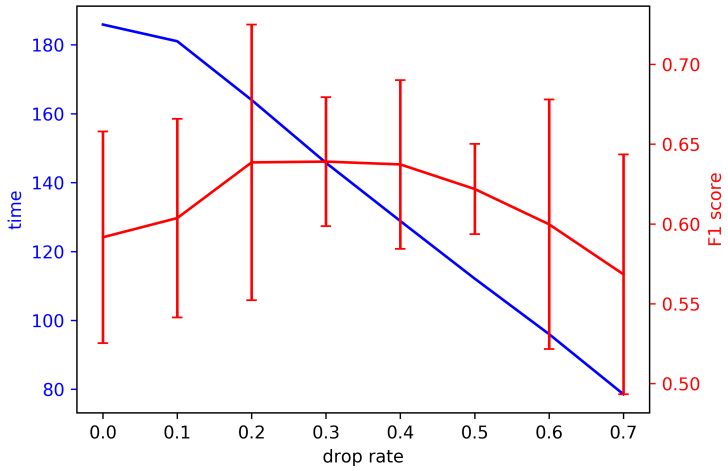

图 3: F1 score 与运行时间

5.3 dropout 参数影响

为了实验效率的提高，我使用算法比较简单的 straddle heuristic 引入 dropout 来进行实验。实验中采样数还是 50 。分别考察不同的 dropout 参数对于最后的 F1 score 和运行时间（5 次重复实验的总时间）的影响，以及观测点分布的影响，结果如图3和图4所示。由图4可以看到，加入 dropout 之后，观测点的分布特征没有很大变化，还是主要分布在阈值边界附近，这也证明了前面所论述的内容，即 dropout 不太会影响到下一个观测点的大致位置。有趣的地方在于，根据图3，虽然时间

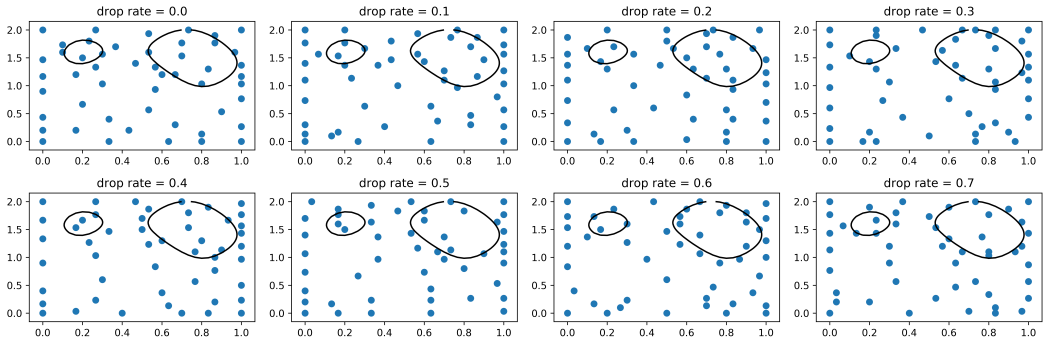

图 4: 观测点位置比较 (dropout)

与预期的一样，基本随着 dropout 率的增加线性减少，但是 F1 score 却是先增后减，也就是说，少量的 dropout 不仅能够降低计算开销，还能够稍微提高准确率，这一点着实令人惊讶。

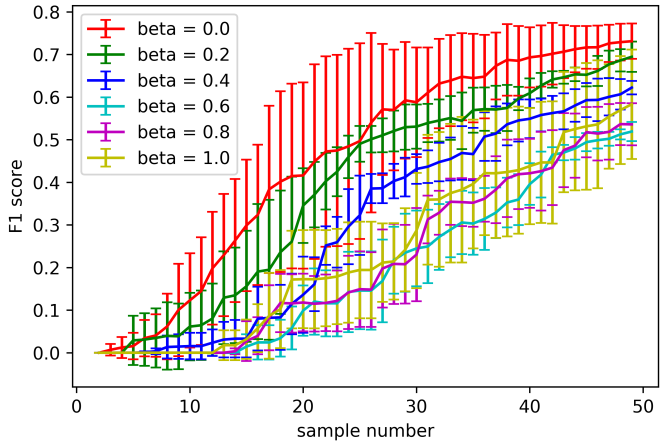

图 5: F1 score 与参数 β 关系

5.4 PV score 参数影响

下面进行实验研究 PV score 的参数 β 的选择对于算法表现的影响，本实验中未引入 dropout 策略，实验结果如图 5 和图 6 。由图5可以看到，$\beta = 0.0$ 的表现最好，略高于 $\beta = 0.2$ 的表现，其他参数设定的表现都要更差一些。我认为这里的原因在于实验中，真实上水平集由两个凸闭集组成，其中一个的元素更多一些，这导致在采样点数相对较少（50）时，模型能够拟合到较大的那个子集，就可以获得相对不错的表现。在这种情况下，$\beta = 0.0$ 的设定使得模型更倾向于“利用”，在当前观测的区域附近进行观测，充分拟合了这个较大的凸集合，从而获得了不错的分数这一点在图 6 中可以看的很清楚，$\beta = 0.0$ 的设定下，模型基本没有观测较小的那个真水平子集附近的点。

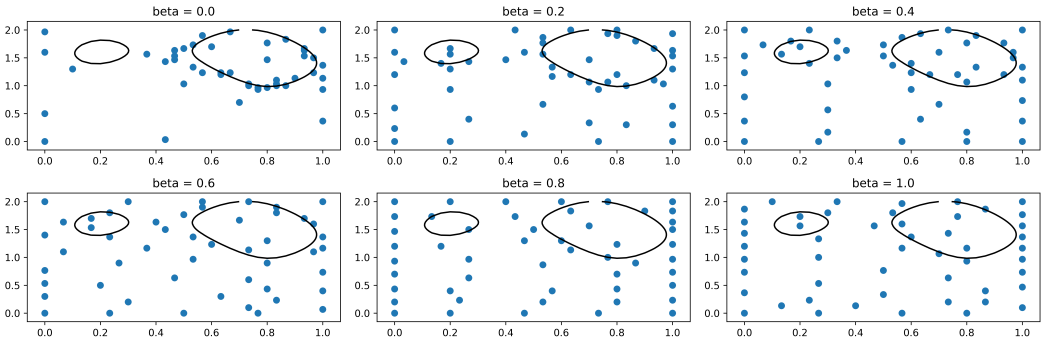

图 6: 观测点位置比较 (PV score)

6 讨论

本文提出了一个新的启发式算法 PV score ，并用实验证明了算法的优异性，与已有工作中的启发式算法相比效果更好。同时，本文分析了该算法中超参数的选择对于模型行为和效果的影响。[1]

另外，本文在 Active Learning 框架下引入 dropout 机制以减少计算复杂度，从理论和实验两个角度证明了其意义。另外，在实验中发现小概率的 dropout 不仅不会对模型表现造成很大的负面影响，还会略微提升其准确率，至于这一点的原因，作者水平有限，并未分析出来，这也是本文的一个遗憾所在。

参考文献

[1] Brent Bryan, Robert C Nichol, Christopher R Genovese, Jeff Schneider, Christopher J Miller, and Larry Wasserman. Active learning for identifying function threshold boundaries. In *Advances in neural information processing systems*, pages 163–170, 2006.

[2] Alkis Gotovos, Nathalie Casati, Gregory Hitz, and Andreas Krause. Active learning for level set estimation. In *IJCAI*, pages 1344–1350, 2013.

[3] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. Springer, 2004.

[4] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[5] Andrea Zanette, Junzi Zhang, and Mykel J Kochenderfer. Robust super-level set estimation using gaussian processes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 276–291. Springer, 2018.

[1] 代码详见见 code 文件夹或者 `https://github.com/Steve-Tod/Super-level-set-estimation`