
Soft Brownian Offset

Release 1.0.5-1-g1684a81

Felix Möller

Aug 25, 2021

CONTENTS:

1	Introduction	1
2	Installation	3
3	Usage	5
4	Background	7
5	Parameter overview	9
6	Gaussian Hyperspheric Offset	11
7	Cite	13

INTRODUCTION

Soft Brownian Offset (SBO) defines an iterative approach to translate points by a most likely distance from a given dataset. It can be used for generating out-of-distribution (OOD) samples. It is based on Gaussian Hyperspheric Offset (GHO), which is also included in this package (see [*below*](#)).

INSTALLATION

This project is hosted on [PyPI](#) and can therefore be installed easily through `pip`:

```
pip install sbo
```

Depending on your setup you may need to add `--user` after the install.

For brevity's sake here's a short introduction to the library's usage:

```
1 from sklearn.datasets import make_moons
2 from sbo import soft_brownian_offset
3
4 X, _ = make_moons(n_samples=60, noise=.08)
5 X_ood = soft_brownian_offset(X, d_min=.35, d_off=.24, n_samples=120, softness=0)
```


BACKGROUND

The technique allows for trivial OOD generation – as shown above – or more complex schemes that apply the transformation of learned representations. For an in-depth look at the latter please refer to the paper that is available [as open access from the CVF](#). For citations please see [cite](#).

PARAMETER OVERVIEW

The following plot gives an overview of possible choices for d_{\min} (d^-), d_{off} (d^+) and softness (σ):

It was created using the following Python code:

```
1  #!/usr/bin/env python3
2  # Creates a plot for Soft Brownian Offset (SBO)
3
4  import numpy as np
5  import pylab as plt
6  import itertools
7  import sys
8
9  from matplotlib import cm
10 from sklearn.datasets import make_moons
11
12 from sbo import soft_brownian_offset
13
14 plt.rc('text', usetex=True)
15
16 c = cm.tab10.colors
17
18 def plot_data(X, y, ax=plt):
19     ax.scatter(X[:, 0], X[:, 1], marker='x', s=20, label='ID', alpha=alpha, c=[c[-1]])
20     ax.scatter(y[:, 0], y[:, 1], marker='+', label='SBO', alpha=alpha, c=[c[-6]])
21
22 def plot_mindist(X, y, ax=plt):
23     if len(X.shape) == 1:
24         X = X[:, None]
25     if len(y.shape) == 1:
26         y = y[:, None]
27     ax.hist(pairwise_distances(y, X).min(axis=1), bins=len(y) // 10)
28     ax.set_xlabel("Minimum distance from ood to id")
29     ax.set_ylabel("Count")
30
31 def plot_data_mindist(X, y):
32     fig, ax = plt.subplots(1, 2)
33     plot_data(X, y, ax=ax[0])
34     plot_mindist(X, y, ax=ax[1])
35     plt.show()
36
37
38 n_samples_id = 60
39 n_samples_ood = 150
```

(continues on next page)

(continued from previous page)

```

40 noise = .08
41 show_progress = False
42 alpha = .6
43
44 n_colrow = 3
45 d_min = np.linspace(.25, .45, n_colrow)
46 softness = np.linspace(0, 1, n_colrow)
47 fig, ax = plt.subplots(n_colrow, n_colrow, sharex=True, sharey=True, figsize=(8.5, 9))
48
49 X, _ = make_moons(n_samples=n_samples_id, noise=noise)
50 for i, (d_min_, softness_) in enumerate(itertools.product(d_min, softness)):
51     xy = i // n_colrow, i % n_colrow
52     d_off_ = d_min_ * .7
53     ax[xy].set_title(f"$d^- = {d_min_: .2f} \ d^+ = {d_off_: .2f} \ \sigma = {softness_}$
54     ↪")
55     if softness_ == 0:
56         softness_ = False
57     y = soft_brownian_offset(X, d_min_, d_off_, n_samples=n_samples_ood,
58     ↪softness=softness_, show_progress=show_progress)
59     plot_data(X, y, ax=ax[xy])
60     if i // n_colrow == len(d_min) - 1:
61         ax[xy].set_xlabel("$x_1$")
62     if i % n_colrow == 0:
63         ax[xy].set_ylabel("$x_2$")
64 ax[0, n_colrow - 1].legend(loc='upper right')
65
66 plt.tight_layout()
67 plt.show()

```

GAUSSIAN HYPERSPHERIC OFFSET

GHO is the basis for SBO and assumes $X \sim \mathcal{N}$. The following code's result displays the shortcomings if the assumption does not hold:

```
1 from sklearn.datasets import make_moons
2 from sbo import soft_brownian_offset, gaussian_hyperspheric_offset
3
4 X, _ = make_moons(n_samples=60, noise=.08)
5 X_ood = (gaussian_hyperspheric_offset(n_samples=220, mu=2, std=.3, n_dim=X.ndim) + X.
  ↳mean()) * X.std()
```


CITE

Please cite SBO in your paper if it helps your research:

```
@inproceedings{MBH21,  
  author    = {Möller, Felix and Botache, Diego and Huseljic, Denis and Heidecker,   
↪Florian and Bieshaar, Maarten and Sick, Bernhard},  
  booktitle = {{Proc. of CVPR SAIAD Workshop}},  
  title     = {{Out-of-distribution Detection and Generation using Soft Brownian   
↪Offset Sampling and Autoencoders}},  
  year      = 2021  
}
```