Supplementary Material for Large Linear Multi-task Gaussian Process Learning

Anonymous Authors[1]

1. Implementation Details

LLGP was implemented in Python 3 from the Anaconda, which offered an Intel MKL-linked scipy (Jones et al., 2001–). The code made heavy use of other packages, namely climin (Bayer et al., 2016), GPy (GPy, since 2012), and paramz (Zwiessele, 2017). Code and benchmarks are available at `<anonymous repository>`.

Application of our approach to all replication studies was carried out on a large server in a multiprogramming environment: CentOS 6.5 with 80 Intel(R) Xeon(R) CPU E7-4870 @ 2.40GHz. The representation evaluation benchmarks were done at once on a cluster of machines running CentOS 5.2-5.9 with Intel(R) Xeon(R) CPU E5430 @ 2.66GHz, where these jobs ran on a single thread per CPU.

References

Bayer, J., Osendorfer, C., Diot-Girard, S., Rckstiess, T., and Urban, Sebastian. climin - a pythonic framework for gradient-based function optimization. http://github.com/BRML/climin, 2016.

GPy. GPy: A Gaussian process framework in python. http://github.com/SheffieldML/GPy, since 2012.

Jones, Eric, Oliphant, Travis, Peterson, Pearu, et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/. [Online; accessed 2017-02-06].

Zwiessele, Max. paramz. https://github.com/sods/paramz, 2017.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.