Journal of Machine Learning Research 13 (2012) 3533-3537 Submitted 4/12; Revised 12/12; Published 12/12

DARWIN: A Framework for Machine Learning and Computer Vision Research and Development

Stephen Gould STEPHEN.GOULD@ANU.EDU.AU
Research School of Computer Science
The Australian National University
Canberra, ACT 0200, Australia

Editor: Antti Honkela

Abstract

We present an open-source platform-independent C++ framework for machine learning and computer vision research. The framework includes a wide range of standard machine learning and graphical models algorithms as well as reference implementations for many machine learning and computer vision applications. The framework contains Matlab wrappers for core components of the library and an experimental graphical user interface for developing and visualizing machine learning data flows.

Keywords: machine learning, graphical models, computer vision, open-source software

1. Introduction

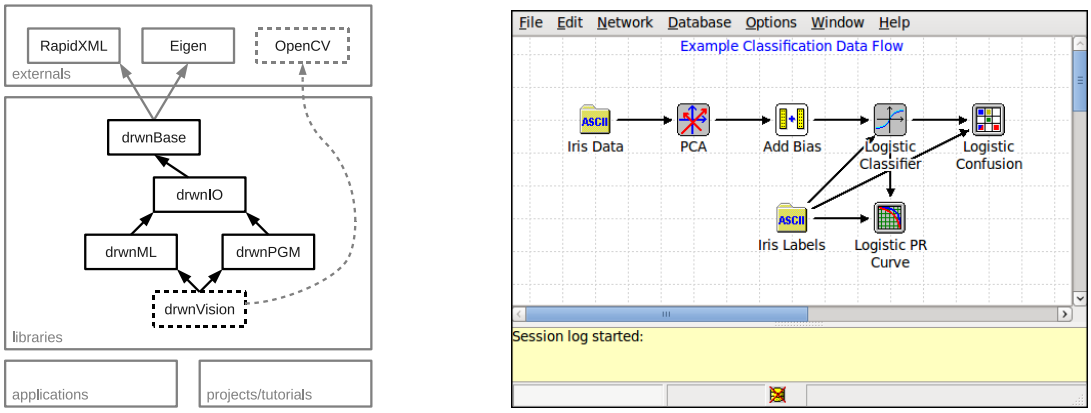
Machine learning and computer vision researchers have benefited over the last few years with the increased availability of numerous high quality open-source toolkits, (e.g., Gashler, 2011; King, 2009; Sonnenburg et al., 2010). These toolkits focus on delivering efficient implementations of mature algorithms but often omit infrastructure necessary for building end-to-end applications or experimenting with new algorithms. Furthermore, domain specific applications, such as those in computer vision, often require functionality from multiple packages complicating integration and maintenance. We have developed the DARWIN framework for machine learning and computer vision that aims to alleviate these limitations.

The goal of the DARWIN framework is two-fold: First, to provide a stable, robust and efficient library for machine learning practitioners, and second, to provide infrastructure for students and researchers to experiment with and extend state-of-the-art methods. To this end, we provide infrastructure for data management, logging and configuration; a consistent interface to standard machine learning and probabilistic graphical models algorithms; and well documented and easy to extend source code. The DARWIN framework is distributed under the BSD license. Importantly, it is free for both academic and commercial use.

2. Libraries

This section describes version 1.4 of the DARWIN framework. The code is divided into five main libraries, which we describe below. Sample applications wrap these libraries to provide out-of-the-box solutions, such classifier training and evaluation or probabilistic inference in graphical models. More sophisticated user applications can easily make use of these libraries.

The libraries have minimal dependence on external codebases, making DARWIN easy to install and maintain. Specifically, we require RapidXML[1] for XML parsing and Eigen[2] for linear algebra. OpenCV (Bradski and Kaehler, 2008) is required only for the (optional) computer vision component of the framework. Applications may link to other libraries to provide additional functionality, for example, GUI interfaces. An illustration of the library components and dependencies is shown in Figure 1(a).



(a) (b)

Figure 1: (a) Illustration of the library components and dependencies of the DARWIN framework. Dashed lines indicate optional components. (b) Screenshot of the experimental GUI for developing machine learning data flows.

Base. The `drwnBase` library provides core infrastructure and utility routines for the rest of the DARWIN framework, including (i) configuration management via command-line arguments and XML file processing, (ii) message, warning and error logging, (iii) runtime code profiling, (iv) basic multi-threading, and (v) miscellaneous routines for string manipulation, filesystem utilities, statistical functions, and XML processing.

IO. Common input/output functionality is provided by the `drwnIO` library. The main component of this library is a persistent data store for managing filesystem storage and in-memory caching of multiple data records.

Machine Learning. The machine learning library, `drwnML`, contains implementations of various algorithms for supervised and unsupervised machine learning. In particular, it includes algorithms for classification, regression and probability distribution modeling.

The library also contains utilities for dimensionality reduction (such as PCA and Fisher's LDA), unconstrained optimization, linear programming, and maintaining sufficient statistics and collating classification results. It also has a sparse vector implementation that is a drop-in replacement for the STL `vector` container. The container can be used to efficiently store sparse vectors with minimal changes to existing code. The concern here was memory usage rather than running time—a naive dot-product implementation, for example, will iterate over all entries in the sparse vector (including the zeros). More efficient dot-product member functions are provided by the sparse vector class.

1. RapidXML found at `http://rapidxml.sourceforge.net`.
2. Eigen found at `http://eigen.tuxfamily.org`.

Probabilistic Graphical Models. The probabilistic graphical models library, `drwnPGM`, provides functionality for (approximate) inference in structured probability spaces over discrete random variables. A factor graph data structure is used to specify the joint distribution in terms of functions, or factors, on overlapping subsets of the variables. The library provides classes for specifying factors and performing common factor operations, such as marginalization. This makes it easy to implement algorithms that manipulate factors. Reference implementations for many inference algorithms are also provided. These include:

- exact inference via the junction tree algorithm (Koller and Friedman, 2009)
- message passing inference algorithms such as synchronous and asynchronous sum-product and min-sum (max-product) (Koller and Friedman, 2009), and sequential tree-reweighted message passing (TRW-S) (Kolmogorov, 2006)
- move-making algorithms such as iterated conditional modes (ICM) (Besag, 1986), and α-expansion and $\alpha\beta$-swap (Boykov et al., 1999)
- linear programming relaxations, (e.g., Komodakis et al., 2007; Globerson and Jaakkola, 2007; Sontag et al., 2008; Meshi and Globerson, 2011)

Our implementations perform favorably against other graphical models inference packages, (e.g., Jaimovich et al., 2010; Mooij, 2010). For example, on the Rosetta Protein Design data set (Yanover et al., 2006), our implementation of the method of Sontag et al. (2008) runs faster than their code[3] on 86.6% of the problems.

Computer Vision. The `drwnVision` library builds on the OpenCV computer vision library (Bradski and Kaehler, 2008) to provide additional functionality for scene understanding. For example, the library provides infrastructure for multi-class semantic segmentation. Many approaches have been proposed to solve this problem. One approach is to construct a conditional Markov random field (CRF) over pixels in the image with learned unary and pairwise terms (He et al., 2004; Shotton et al., 2006). Another approach is to perform non-parametric label transfer by matching into a corpus of annotated images, (e.g., Liu et al., 2009), and does not rely on discriminatively trained classifiers. The vision library provides instances of both approaches and XML configurable applications make it easy to train and evaluate on new data sets. Moreover, documentation and scripts for generating results on standard data sets are distributed with the library.

Briefly, our CRF implementation is an enhanced variant of the baseline method described in Gould et al. (2009). It includes learned boosted decision tree classifiers for the unary terms and cross-validated contrast sensitive pairwise terms. On the standard 21-class MSRC data set (Criminisi, 2004) we achieve 78.4% accuracy using only unary terms and 82.3% with the full CRF model; on the Stanford Background data set (Gould et al., 2009) we achieve 77.1% (91.3%) accuracy and 79.9% (92.6%) accuracy on the semantic (geometric) labeling task for the unary and CRF model, respectively.

Our label transfer implementation makes use of the PatchMatch algorithm (Barnes et al., 2009) to construct a graph of dense patch correspondences between images. A full description of this approach is provided in Gould and Zhang (2012). Here we achieve 64.9% and 69.6% for the MSRC and Stanford Background data sets, respectively.

3. Code found at `http://cs.nyu.edu/~dsontag/code/mplp_ver1.tgz`.

3. Projects and Applications

The DARWIN framework is distributed with source code for a number of applications and projects that expose the functionality of the library, for example, training and evaluating classifiers or performing inference in probabilistic graphical models. Matlab `mex` wrappers are also provided for accessing core algorithms within the framework from the Matlab environment.

For non-expect machine learning practitioners it is often helpful to provide a graphical environment for designing machine learning pipelines. To this end, we have developed an experimental graphical user interface (GUI) for constructing machine learning data flows. A screenshot of the GUI is shown in Figure 1(b). The GUI is built using wxWidgets,[4] which must be installed separately. We intend to continue developing the GUI in future releases.

4. Documentation

Comprehensive documentation, including download and installation instructions, API reference, and short tutorial is available online at `http://drwn.anu.edu.au`. The documentation can also be generated locally from the source code using Doxygen.[5]

Acknowledgments

Some components of DARWIN were derived from the STAIR VISION LIBRARY[6] developed by the same author while at Stanford University. We thank the many contributors to that library. We also thank the early users of DARWIN for their suggestions and bug reports.

References

C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *SIGGRAPH*, 2009.

J. E. Besag. On the statistical analysis of dirty pictures. *Royal Stat. Soc. Series B*, 48:259–302, 1986.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV*, 1999.

G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

A. Criminisi. Microsoft research cambridge (MSRC) object recognition pixel-wise labeled image database (version 2), 2004. Available for download under license from MSR at `http://research.microsoft.com/en-us/projects/ObjectClassRecognition/`.

M. S. Gashler. Waffles: A machine learning toolkit. *JMLR*, pages 2383–2387, July 2011.

4. wxWidgets found at `http://www.wxwidgets.org`.
5. Doxygen found at `http://www.doxygen.org`.
6. STAIR Vision Library found at `http://ai.stanford.edu/~sgould/svl`.

A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *NIPS*, 2007.

S. Gould and Y. Zhang. PatchMatchGraph: Building a graph of dense patch correspondences for label transfer. In *ECCV*, 2012.

S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

X. He, R. S. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

A. Jaimovich, O. Meshi, I. McGraw, and G. Elidan. FastInf: An efficient approximate inference library. *JMLR*, 11:1733–1736, 2010.

D. E. King. Dlib-ml: A machine learning toolkit. *JMLR*, 10:1755–1758, 2009.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 2006.

N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *ICCV*, 2007.

C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, 2009.

O. Meshi and A. Globerson. An alternating direction method for dual MAP LP relaxation. In *ECML*, 2011.

J. M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *JMLR*, 11:2169–2173, Aug 2010.

J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN machine learning toolbox. *JMLR*, 11:1799–1802, June 2010.

D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, 2008.

C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation—an empirical study. *JMLR*, 2006.