Journal of Machine Learning Research 14 (2013) 3159-3163 Submitted 6/13; Revised 8/13; Published 10/13

Divvy: Fast and Intuitive Exploratory Data Analysis

Joshua M. Lewis JOSH@COGSCI.UCSD.EDU
Virginia R. de Sa DESA@COGSCI.UCSD.EDU
Department of Cognitive Science
University of California, San Diego
La Jolla, CA 92093-0515, USA

Laurens van der Maaten LVDMAATEN@GMAIL.COM
Faculty of Elect. Eng., Math., and Comp. Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Editor: Mark Reid

Abstract

Divvy is an application for applying unsupervised machine learning techniques (clustering and dimensionality reduction) to the data analysis process. Divvy provides a novel UI that allows researchers to tighten the action-perception loop of changing algorithm parameters and seeing a visualization of the result. Machine learning researchers can use Divvy to publish easy to use reference implementations of their algorithms, which helps the machine learning field have a greater impact on research practices elsewhere.

Keywords: clustering, dimensionality reduction, open source software, human computer interaction, data visualization

1. Introduction

The field of machine learning has produced many techniques for performing data analysis, but researchers outside the field face substantial challenges applying them to their data. First, new techniques are difficult to access. Authors often describe an algorithm without providing a reference implementation, and if they do provide code it may be for an unfamiliar language or platform. Second, new techniques are difficult to apply correctly. A new technique might make strong assumptions about the structure of its input or be very sensitive to parameter changes. Third, in the early, exploratory stage of data analysis researchers should explore and compare several different techniques. If each technique is challenging to get running for the reasons above, the challenge is compounded with multiple techniques using different languages and data formats.

We have built Divvy to ameliorate these problems for those who would like to use unsupervised machine learning techniques in their research (specifically clustering and dimensionality reduction), and provide a platform for machine learning researchers to publish fast, easy to use versions of their algorithms. Using Divvy, researchers can quickly run an assortment of clustering and dimensionality reduction algorithms on their data, without worrying about programming languages, data formats, or visualization.

Divvy is a member of a family of attempts to bring machine learning and data visualization to a wider audience. The GGobi project (Swayne et al., 2003) provides a variety of interactive visualiza-

tions for high-dimensional data, and includes some lightweight machine learning components such as PCA. The Orange project (University of Ljubljana Bioinformatics Laboratory, 2013) provides a visual programming interface for machine learning techniques in Python. In the commercial sector, Ayasdi, Inc. uses topology to help customers analyze their data intuitively (Ayasdi, Inc., 2013). Divvy's unique focus among these projects is on user experience and extensibility.

For users, Divvy provides a simple, fast interface for doing data analysis. For machine learning researchers, Divvy provides a plugin architecture that lets one release a version of an algorithm complete with custom UI and help resources. By publishing an algorithm on the Divvy platform, machine learning researchers can drastically lower the barriers to entry that data analysts face when attempting to use it. Divvy and all of its included plugins are open source, distributed under the MIT license.

2. Interaction

Divvy has three fundamental components to its interface (see Figure 1). In the lower-right corner of the main window (Label 1) is a collection of data sets for the user to analyze. Each data set is associated with one or more data set views, which are visualized in the left hand portion of the interface (Label 2). Finally each data set view is characterized by a combination of four plugins from the top-right panel (Label 3): a dimensionality reduction algorithm, a clustering algorithm, a point visualizer, and a data set visualizer.

Clustering and dimensionality reduction plugins are interfaces to their associated algorithms, currently K-means and single/complete linkage for clustering, and PCA, Isomap (Tenenbaum et al., 2000) and t-SNE (van der Maaten and Hinton, 2008) for dimensionality reduction. Point visualizers render data points in a meaningful way, for example, by rendering images in a data set where points represent images, or rendering type for a linguistic corpus. Data set visualizers render the entire data set, for example as a scatter plot or parallel coordinates plot. Divvy includes an image point visualizer and a scatter plot data set visualizer by default.

As an example, the user in Figure 1 has loaded four data sets into Divvy and selected the faces data set. The user has created three views that represent three distinct perspectives on the data. The top-right view is an Isomap embedding of the data set with an image point visualizer, so the user sees a selection of points rendered as images and positioned in the first two Isomap dimensions. The lower left view is a t-SNE embedding with coloring from K-means clustering rendered as a scatter plot.

In practice a researcher can use these visualizations to evaluate different approaches to dimensionality reduction and clustering. In Figure 1 the user can easily see that Isomap embeds the face images in two dimension more smoothly than t-SNE. The structure of the faces data set is ideal for Isomap, and a researcher can quickly discover that with Divvy.

Users can create as many perspectives on their data as they'd like, grow or shrink them with the slider on the lower-right edge of the screen, and export their preferred views as PNGs. Whenever they change a plugin or plugin parameter, the selected view automatically rerenders with the new setting. In order to make Divvy as responsive as possible, each data set view is sandboxed in its own set of threads (Divvy is task parallel with granularity at the view level). Users can start a long computation in one view while still interacting with other views or data sets. For shorter computations the view changes instantly, giving users immediate visual feedback on the effect of their parameter selections.

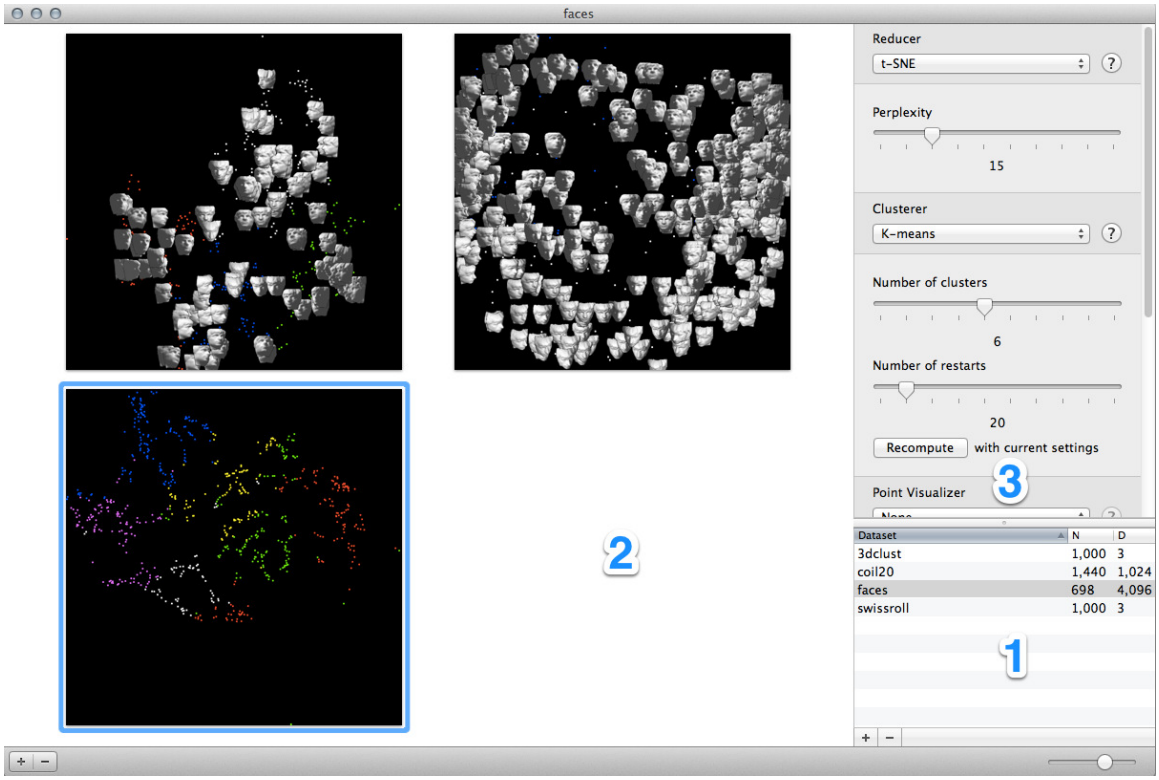

Figure 1: Divvy's UI. (1) The data sets list—data sets the user has loaded appear here with some summary statistics. (2) The data set view palette—data set views are visualizations of data sets using a combination of a dimensionality reduction algorithm, a clustering algorithm, a point visualizer, and a data set visualizer. Each data set can have multiple data set views. (3) Data set view parameters—controls for setting the parameters of the algorithms that compose a particular data set view, for example, the number of clusters for a clustering algorithm.

Our goal with the Divvy UI is to tighten up the action-perception loop in data analysis. As an analogy, baseball players have an excellent idea of how baseballs behave. A baseball's behavior is, of course, governed by the laws of physics and an explicit description of that behavior might be quite complex when spin, deformation, wind and field texture are taken into account. Nevertheless, through extensive experience baseball players acquire an excellent pragmatic understanding of how baseballs behave, an understanding that one might guess is based on an implicit learned model of baseball behavior rather than the explicit model a physicist would give. By giving Divvy users an immediate, tactile experience of algorithmic behavior, we hope they can develop better intuitive models of how algorithms behave and thus make better analysis decisions.

3. Architecture

The core Divvy application performs no computation. Rather it is a lightweight framework for loading data sets and plugins and then coordinating their interaction. Each plugin is an independent bundle that defines a UI and follows one of four possible input/output protocols: clusterer, reducer, point visualizer and data set visualizer.

While the core Divvy application is Mac OS X specific, each machine learning plugin is just a lightly-wrapped reference implementation of its algorithm in C or C++.[1] A researcher publishing an algorithm in Divvy's format need only add an OS X UI (and Divvy gives complete freedom as to the details of that UI save a fixed width) to their implementation, which remains completely platform-agnostic. With this bit of work users can drop the algorithm bundle into Divvy and start using the new technique on their existing data sets.

We implemented Divvy on Mac OS X in order to focus on one user experience. While it's of course desirable to have open source software on as many platforms as possible, building the Divvy UI across platforms was outside the scope of our engineering resources.[2]

Divvy can import data from CSV or a simple BIN format, and we have released R to Divvy and MATLAB to Divvy exporters. Data within Divvy can be exported as PNG or CSV as appropriate. Our goal is that Divvy can fit well into diverse analysis workflows.

4. Performance

Divvy is both task and data parallel. As mentioned above, each data set view owns a set of threads that compute independently from the UI and those of other views. Within a view each plugin can operate in parallel over its assigned data set. In our lab Divvy can achieve over 2,300% CPU utilization on our hyperthreaded 12-core Mac Pro through a combination of these two forms of parallelization.[3]

The task parallelism is achieved at the application level through the NSOperation framework in Cocoa. Plugin authors get it for free. Data parallelism is the responsibility of plugin authors and should be implemented using the open-source libdispatch library.[4]

Acknowledgments

We would like to acknowledge support for this project from the National Science Foundation (NSF grant SES-0963071).

References

Ayasdi, Inc. Iris: Query-free insight discovery, 2013. URL `http://www.ayasdi.com/product/`.

1. Overall the Divvy codebase is two-thirds Objective-C (interface, plugin wrappers) and one-third C/C++ (algorithms). As algorithms are added, this balance will shift towards platform-independent code. To port Divvy to another platform, one would only need to rewrite the interface code.
2. Joshua Lewis wrote the entire Divvy application and bundled plugins save the dimensionality reduction plugins, which were written by Laurens van der Maaten.
3. With twelve cores and two threads per core, peak utilization is 2,400%.
4. We picked libdispatch because OpenMP has an issue with GCC 4.2/4.3 where starting a parallel section from a thread other than the main thread causes a crash, so we cannot recommend it.

Deborah F. Swayne, Duncan Temple Lang, Andreas Buja, and Dianne Cook. GGobi: evolving from XGobi into an extensible framework for interactive data visualization. *Computational Statistics & Data Analysis*, 43:423–444, 2003.

Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000.

University of Ljubljana Bioinformatics Laboratory. Orange - data mining fruitful and fun, 2013. URL `http://orange.biolab.si`.

Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.