Denali: A tool for visualizing scalar functions as landscape metaphors

Justin Eldridge
Mikhail Belkin
Yusu Wang

September 21, 2014

Abstract

Many sources of data in machine learning, such as probability distributions and cost functions, can be summarized by extracting tree-like structures. These trees, however, are often large enough so as to be difficult to visualize with traditional methods. In this paper, we present *denali*, an interactive interface with novel features for visualizing *landscape metaphor* representations of trees, allowing for the intuitive exploration of large and complex data sets. Our software and additional examples of its usage can be found at `http://denali.cse.ohio-state.edu`.

1 Introduction

Many sources of data – from complex networks to high-dimensional probability distributions – can be represented or summarized by trees. Visualizing such trees can yield important insights into the structure of the data. However, the traditional approach of plotting the tree as a graph in the plane becomes unwieldy when the number of nodes is large. Furthermore, it is often the case that one wishes to visualize not only the structure of a tree, but also one or multiple attributes associated with each node (say, the age and the salary of a person represented by a node). Again, traditional methods offer limited opportunities to do so.

Denali is a tool for visualizing trees as *landscape metaphors* – mountain-like 2D surfaces which provide a natural means for visualizing both the structure of a tree and scalar attributes defined on top of it [6, 4]. *Denali* allows data sets that would be cumbersome to visualize as graphs in the plane to be visually parsed and manipulated in a fully interactive and intuitive manner.

The conceptual "pipeline" of using *denali* is shown in Figure 1(a). *Denali* focuses on the last step: from a scalar tree to a terrain visualization. Additionally, the included utilities aid the user in the "extraction" step, allowing him or her to apply *denali* to many different sources of data. For example, by using the included *ctree* tool, the user can extract and represent the topological structure of a scalar function defined on a point cloud as a *contour tree* [1], which can then be visualized as a landscape metaphor.

In [3], Harvey et al. developed a visualization platform, called Ayla, to explore high dimensional molecular simulation data based on the terrain metaphor idea. In particular, the contour tree of the potential energy of protein conformations is constructed from input molecular simulation data. While Ayla is specifically designed for analyzing molecular simulation data, we aim to provide a generic tool, enabling users to apply it to their specific tasks at hand. Some examples of using our tool to analyze diverse data sources can be found at `http://denali.cse.ohio-state.edu#gallery`. The software download includes demos and documentation which demonstrate the diverse

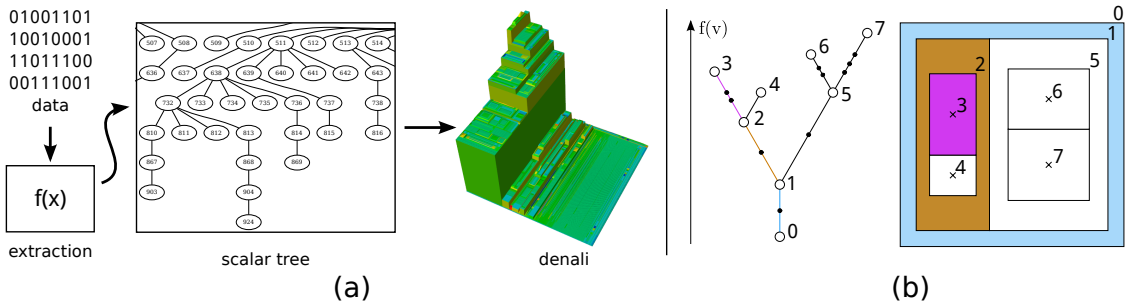

Figure 1: a) The conceptual "pipeline" used when working with *denali*. The visualization is of a comment thread from an online community. The scalar tree shown is cropped for clarity of presentation. b) The correspondence between arcs in the scalar tree and the areas between contours in the landscape.

usage of *denali* and its associated tools, meaning that integrating the software into one's workflow is simple and straightforward.

Denali introduces several novel features designed to ease the analysis of complex data sets from various domains. The interactive local simplification tool allows the user to find trends in noisy data sets. The custom colormap functionality permits the visualization of two scalar functions simultaneously. And the powerful callback system enables the user to tailor the behavior of *denali* to his or her specific requirements.

In this paper, we describe the concept of a landscape metaphor by a motivating example and outline the novel features that *denali* introduces. We also provide several examples of *denali*'s applications to demonstrate its usefulness in machine learning and data analysis.

2 Visualizing a comment thread as a landscape metaphor

The last step of Figure 1(a) shows an example of *denali* as applied to visualize an online comment thread. Our data is a set of 954 comments from a threaded discussion. Each comment (if it is not a top-level comment) has a parent comment and zero or more children comments. Additionally, every comment has an associated integer score resulting from the users assigning up-votes and down-votes. For reasons that will become clear later, we recursively assign to each node an associated scalar value computed by summing its comment score and it's parent's computed scalar value – that is, if the root node has score 10, and a child of the root has score 5, then the value associated with the root will be 10, while the value associated with the child will be 15.

The resulting structure – a tree for which every node has an associated scalar value – is referred to as a *scalar tree*. *Denali* requires that its input be in the form of a scalar tree, but, as previously mentioned, tools are provided to extract scalar trees from other sources of data, such as scalar functions defined on point clouds. Part of the scalar tree generated by the comment thread data set was plotted using *graphviz* [2] and shown in Figure 1(a). If the entire output were pictured at this same resolution, the image would be 20 inches wide. In comparison, *denali*'s visualization is compact and clear.

The visualization produced by *denali* and shown in Figure 1(a) is referred to as the *landscape metaphor* representation of the scalar tree. Viewed from above, the landscape appears as a set of nested rectangles and points. Each node in the tree is mapped to a feature; either a rectangular feature, if the node is a branch node, or a point, if the node is a leaf. As a special case, the root node is mapped to the outermost rectangle. The features corresponding to the children of the root are recursively mapped to lie inside the root rectangle. Figure 1(b) makes this clear.

The rectangular features vary in area such that the size of a rectangle corresponding to a given

node represents the number of nodes in the subtree rooted at that node. Therefore, in this case, the size of a rectangular feature denotes the number of comments in the subdiscussion spawned by that comment. Furthermore, as Figure 1(b) shows, the area between two rectangular features corresponds to an arc of the tree. In general, arcs may have zero or more *member nodes* which carry a weight so that a total weight can be assigned to the arc. The area between features is chosen to be proportional to this total weight. In the case of the current data set no arc has members, and so there is no space between rectangular features. In comparison, traditional methods of plotting the tree offer limited ways to depict the "size" of a component.

Viewed again from the side, each rectangle and point in the landscape does not exist in the plane, but is rather "lifted" in the third dimension to a height corresponding to its node's associated scalar value. We can therefore interpret the height of a feature as the popularity of a comment in the thread. We have chosen the scalar value in such a way that popular sub-discussions are more prominent.

We can now begin to interpret the visualization. Top-level comments appear left-to-right in the same order as they did on the page they were collected from. The first comment corresponds to the large rectangular block at the left of the landscape. As the size of the block corresponds to the number of comments in the subthread root at the comment, it is instantly clear that replies in the subdiscussion spawned by this comment contribute about one quarter of all comments in the thread.

We have also supplied a custom coloring function which conveys the "agreeability" of each comment. This is done by mapping a comment's upvote-to-downvote ratio to a color between blue and red. Red comments have a high upvote-to-downvote ratio, suggesting that they are popular and agreeable. Comments with a low ratio are colored blue. Visualizing the comment scores via height and the agreeability index as color allows us to instantly identify a comment which had a very high agreeability ratio, but which was not highly-upvoted. Further inspection showed that this comment was very technical and full of factual information, to the point that is was perhaps inaccessible to the layman. On the other hand, the most popular comment was more subjective and opinion-based; easy to understand, but perhaps more controversial because of its nature. We believe that this sort of insight would have been difficult to make if the tree were plotted as a graph in the plane.

3 Features and Implementation

Denali includes several novel features which allow it to be used for many different sources of data, including noisy trees, and domain-specific applications.

- *Local interactive simplification* - Trees extracted from noisy data can often have many spurious features. Simplification is one approach to suppressing these features. To our knowledge, *denali* is the first software to provide fully interactive and local simplification tools, so that the user may select a specific region of the landscape – for example, the area around a local minimum in a cost function – and visualize it at a level of simplification independent of the rest of the landscape.
- *Custom color function* - The user can specify a second scalar attribute on each node and member of the tree. This attribute is used to color the landscape. This is useful for visualizing the behavior of two related functions and identifying where they differ.
- *Powerful callback system* - *Denali* provides a callback system that allows custom code to be invoked whenever the user makes a selection. The user can write her code in any language.

The callback system is very general. When visualizing a probability distribution, for example, the user may invoke a callback to interactively resample the function in a region and automatically update the contour tree, allowing her to increase the number of samples in areas of interest.

- *Other utilities* - *Denali* includes several utilities to help streamline working with the software. The *ctree* tool computes contour trees from graph data, and the included Python package provides functions for reading *denali* formats and computing contour trees from point cloud data.

Demos. *Denali* comes complete with several working demos which serve as examples of how to use the software. To see screenshots of each of the demos in action, visit `http://denali.cse.ohio-state.edu/#gallery`.

- *Internet comment thread* - The visualization is described in Section 2. Clicking on on a point in the landscape invokes a callback function that prints the corresponding comment(s) to the screen.
- *Hierarchical clustering* - Images of handwritten digits are clustered using single-linkage clustering. Selecting a component plots an average image of the digits in the cluster.
- *Neural network model space* - A neural network is trained on samples of the sine function and the cost landscape is visualized. Selecting a component evaluates the model and plots its output.
- *Probability distribution* - A mixture of Gaussians in four dimensions is visualized. Selecting a part of the landscape automatically generates more samples in that region and regenerates the visualization.

Implementation. *Denali* is implemented in C++ using the open-source VTK [5] and Qt (`http://qt-project.org/`) toolkits. It runs on both Linux and Windows. It is released under an open-source BSD license.

4 Conclusion

Denali is a powerful tool which enables users to visualize complex tree structures which may be unwieldy to visualize by more traditional means. With the included extensions, scalar functions on point clouds and graphs can also be visualized. For more information, documentation, and a tutorial, see the *denali* website at `http://denali.cse.ohio-state.edu`.

References

[1] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 918–926. Society for Industrial and Applied Mathematics, 2000.

[2] E. R. Gansner and S. C. North. An open graph visualization system and its applications to software engineering. *Software - Practice and Experience*, 30(11):1203–1233, 2000.

[3] W. Harvey, I.-H. Park, O. Rbel, V. Pascucci, P.-T. Bremer, C. Li, and Y. Wang. A collaborative visual analytics suite for protein folding research. *Journal of Molecular Graphics and Modelling*, 53:59–71, Sept. 2014.

[4] W. Harvey and Y. Wang. Topological landscape ensembles for visualization of scalar-valued functions. In *Computer Graphics Forum*, volume 29, pages 993–1002. Wiley Online Library, 2010.

[5] W. Schroeder, K. M. Martin, and W. E. Lorensen. *The Visualization Toolkit (2Nd Ed.): An Object-oriented Approach to 3D Graphics*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1998.

[6] G. H. Weber, P.-T. Bremer, and V. Pascucci. Topological landscapes: A terrain metaphor for scientific data. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1416–1423, 2007.