Bioinformatics, 32(16), 2016, 2562–2564
doi: 10.1093/bioinformatics/btw196
Advance Access Publication Date: 13 April 2016
Applications Note

OXFORD

Data and text mining

Muxstep: an open-source C++ multiplex HMM library for making inferences on multiple data types

Petar Veličković* and Pietro Liò

Computer Laboratory, University of Cambridge, Cambridge CB3 0FD, UK

*To whom correspondence should be addressed.

Associate Editor: Jonathan Wren

Received on February 10, 2016; revised on March 21, 2016; accepted on April 6, 2016

Abstract

Motivation: With the development of experimental methods and technology, we are able to reliably gain access to data in larger quantities, dimensions and types. This has great potential for the improvement of machine learning (as the learning algorithms have access to a larger space of information). However, conventional machine learning approaches used thus far on single-dimensional data inputs are unlikely to be expressive enough to accurately model the problem in higher dimensions; in fact, it should generally be most suitable to represent our underlying models as some form of complex networksng;nsio with nontrivial topological features. As the first step in establishing such a trend, we present **muxstep**, an open-source library utilising multiplex networks for the purposes of binary classification on multiple data types. The library is designed to be used out-of-the-box for developing models based on the multiplex network framework, as well as easily modifiable to suit problem modelling needs that may differ significantly from the default approach described.
Availability and Implementation: The full source code is available on GitHub: https://github.com/PetarV-/muxstep
Contact: petar.velickovic@cl.cam.ac.uk
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Machine learning algorithms that tackle problems such as classification, clustering and decoding have been frequently found in a variety of applications, a significant share of which is of bioinformatical nature. Larrañaga *et al.* (2006) have presented a review of these applications, which includes but is not limited to knowledge extraction, genomics and proteomics, systems biology and text mining. In a variety of fields, perhaps especially bioinformatics, the algorithms can be faced with learning from *multiple types of data simultaneously*. The exact ways in which the processes producing each data type are connected may well be unknown or incompletely understood, which often results in having to make certain assumptions on their form (to make the problem tractable).

One possible assumption is to model these 'inter-process connections' as multilayer networks (Kivelä *et al.*, 2014); in particular, the special case of *multiplex networks* could be very appropriate while still maintaining tractability. Informally, a multiplex network is a multi-layered graph in which each layer is built over the same set of nodes, and there may exist edges between nodes in different layers. This models. suitability arises from the fact that there exists a wide variety of systems exhibiting 'natural' multiplexity: social networks and epidemics modelling (Granell *et al.*, 2014), transportation networks (Cardillo *et al.*, 2013), biochemical and genetic networks (Stark *et al.*, 2006) and research communities (De Domenico *et al.*, 2015), to name a few.

We have developed the **muxstep** library motivated by the examples above, as the (to the best of our knowledge) first publicly available machine learning library acquiring this methodology. The problem we focus on is the very common problem of *binary classification* (the Supplementary Materials also detail workflows that

extend it to *k*-ary classification). Briefly, **muxstep** is designed to process any ordered sequence of data points, where each data point is actually a tuple of several real-valued data types. Furthermore, it is possible to assign a discrete '*sub-output*' to each tuple in the sequence, thus encompassing a kind of mixture model.

The **muxstep** library and this applications note are designed with the following ***primary*** *goals* in mind: (i) *Out-of-the-box usefulness* for new problems, and providing an overview of its essential internals and a worked case study in a way that makes it easy for bioinformaticians to develop models based on multilayer networks; (ii) *Ease of extending* (in the spirit of open-source), with clear descriptions of the most common ways to extend it; (iii) *User-friendliness and potential educational value* of its codebase, with a well-structured coding style and an abundance of comments and explanations of key algorithms involved.

2 Implementation

The **muxstep** library is implemented in C++ and is not dependent on any libraries outside of the C++11 standard template library. In order to build the library from source (or link it against other C++ code), clang++ is required; on Mac OS X, this compiler is installed by default.

The library is capable of *four* ***essential*** *actions*: (i) *Creating (randomised) models* with a given number of nodes, or *copying existing models*; (ii) *Training the models* on a set of sequences for which it is known whether they belong in the first (*positive*) or the second (*negative*) class; (iii) *Classifying new (unseen) sequences* using a trained model; (iv) *Storing the model parameters* for later use.

The primary tasks of training and classification are performed by using established algorithms; the *Baum-Welch algorithm* is used to train individual Gaussian mixture hidden Markov model (GMHMM) layers based on each data type separately. Afterwards, *NSGA-II* by Deb *et al.* (2002) is used to train the interlayer transition weights between them. Finally, treating the entire structure as a large-scale GMHMM, the *forward algorithm* is used to evaluate likelihoods for new sequences to be used for classification. The full pipeline is illustrated by Figure 1.

We have fully described the theoretical foundations of the model (as well as an application to classifying patients for *breast invasive carcinoma*) in Veličković and Liò (2015) important details of this case study are outlined, for convenience, in the supplementary data.

3 Usage

Within the repository, we have provided an example C++ source file illustrating the basic workflow while making use of the **muxstep** library. The source is located in example/folder, along with auxiliary data used for training and testing within the example. The example is well-commented and covers a method for extracting the training and testing data into a data structure suitable for the classifier, creating, training and classifying on the model, and methods for storing its parameters in a file for later use, as well as creating a model from such a file.

The supplementary data contains a full manual on how to install and use **muxstep**, including with it the detailed guidelines on how to modify parts of the library to suit particular requirements. These include, but are not limited to: (i) Disabling the use of sub-outputs; (ii) Using output distributions different than the Gaussian distribution;

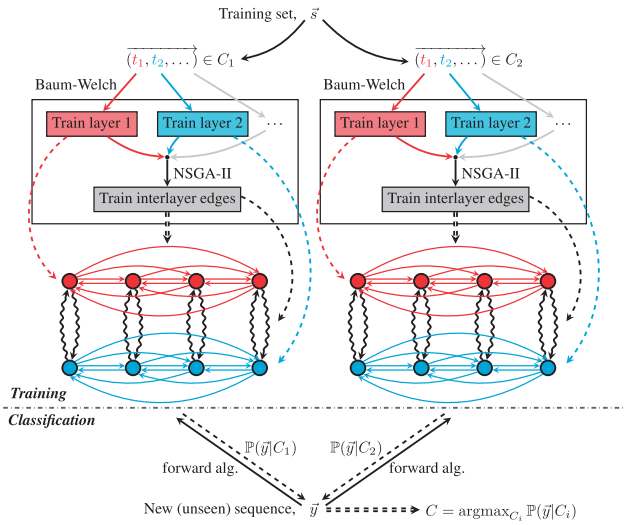

Fig. 1. A diagram representing all the steps in the training and classification algorithms The training algorithm initially partitions the training set $\vec{s}$ into sequences belonging to class C_1 and those belonging to class C_2. Then it produces two models (one trained on all the sequences in C_1, the other on all the sequences in C_2). The models are produced by first training the individual layers on the sequences containing the individual data types only (e.g. for a sequence [(1, 2), (3, 3)], the layers will be trained on sequences [1, 3] and [2, 3], respectively). Then the layers are connected with appropriate interlayer transition weights using NSGA-II. The classification algorithm queries these two models for the likelihood $\mathbb{P}(\vec{y}|C_i)$ of producing an unseen sequence, $\vec{y}$, and chooses the class C corresponding with the model more likely to have produced the sequence. **N.B.** this can easily and intuitively be extended to more than two classes; see the supplementary data for more information (Color version of this figure is available at *Bioinformatics* online.)

(iii) Computing confidence intervals for the obtained output sequence likelihoods.

For convenience, we also provide two auxiliary tools within the test/folder which should be helpful with testing the generated models:

- A *synthetic data generator* (contained within test/syn_gen/), which can generate normally distributed sequence data using a given parameter file.
- An *evaluation suite* (contained within test/classifier/), capable of performing *k*-fold crossvalidation on a set of sequences with known labels to evaluate the model's performance by computing various metrics such as the accuracy, sensitivity, MCC, F_1 score etc. It is also capable of performing *noise testing*, evaluating the robustness of the model in the presence of Gaussian noise on the data.

4 Conclusion

In this applications note we have presented **muxstep,** a C++ library for binary classification on multiple data types, taking advantage of hidden Markov models and multiplex networks. To the best of our knowledge, it is the first of its kind, and should serve as both a valuable tool for bioinformaticians (as well as other scientists in a variety of fields) and a foundation for future advancements in the area of multi-omic data integration. The library has been designed not only to be simple to use as compiled from source, but also to be easily extensible. We intend to continue developing, extending and maintaining it, and we hope that the work described here will represent an important step towards a new class of software implementations within the context of complex multilayer networks, which should be

highly beneficial to users needing to properly handle multiple types of data simultaneously.

The full source code of **muxstep** is freely available on the corresponding author's GitHub profile and licensed under the MIT license.

Funding

We would like to thank the EU grant EpiHealthNet.

Conflict of Interest: none declared.

References

Cardillo,A. *et al*. (2013) Emergence of network features from multiplexity. *Scientific Reports*, **3**, 1344.

De Domenico,M. *et al*. (2015) Identifying modular flows on multilayer networks reveals highly overlapping organization in interconnected systems. *Physical Review X*, **5**.

Deb,K. *et al*. (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, **6**, 182–197.

Granell,C. *et al*. (2014) Competing spreading processes on multiplex networks:awareness and epidemics. *Physical Review E*, **90**, 1344.

Kivelä,M. *et al*. (2014) Multilayer networks. *Journal of Complex Networks*, **2**, 203–271.

Larrañaga,P. *et al*. (2006) Machine learning in bioinformatics. *Briefings in Bioinformatics*, **7**, 86–112.

Stark,C. *et al*. (2006) BioGRID: a general repository for interaction datasets. *Nucleic Acids Research*, **34** (suppl. 1), D539–D539.

Veličković,P. and Liò,P. (2015) Molecular multiplex network inference using Gaussian mixture hidden Markov models. *Journal of Complex Networks Advance Access*, doi: 10.1093/comnet/cnv029.