Journal of Machine Learning Research 19 (2018) 1-5 Submitted 4/18; Published 10/18

Scikit-Multiflow: A Multi-output Streaming Framework

Jacob Montiel JACOB.MONTIEL@TELECOM-PARISTECH.FR
LTCI, Télécom ParisTech, Université Paris-Saclay
Paris, FRANCE

Jesse Read JESSE.READ@POLYTECHNIQUE.EDU
LIX, École Polytechnique
Palaiseau, FRANCE

Albert Bifet ALBERT.BIFET@TELECOM-PARISTECH.FR
LTCI, Télécom ParisTech, Université Paris-Saclay
Paris, FRANCE

Talel Abdessalem TALEL.ABDESSALEM@ENST.FR
LTCI, Télécom ParisTech, Université Paris-Saclay
Paris, FRANCE
UMI CNRS IPAL, National University of Singapore

Editor: Balazs Kegl

Abstract

scikit-multiflow is a framework for learning from data streams and multi-output learning in Python. Conceived to serve as a platform to encourage the democratization of stream learning research, it provides multiple state-of-the-art learning methods, data generators and evaluators for different stream learning problems, including single-output, multi-output and multi-label. scikit-multiflow builds upon popular open source frameworks including scikit-learn, MOA and MEKA. Development follows the FOSS principles. Quality is enforced by complying with PEP8 guidelines, using continuous integration and functional testing. The source code is available at `https://github.com/scikit-multiflow/scikit-multiflow`.

Keywords: Machine Learning, Stream Data, Multi-output, Drift Detection, Python

1. Introduction

Recent years have witnessed the proliferation of Free and Open Source Software (FOSS) in the research community. Specifically, in the field of Machine Learning, researchers have benefited from the availability of different frameworks that provide tools for faster development, allow replicability and reproducibility of results and foster collaboration. Following the FOSS principles, we introduce scikit-multiflow, a Python framework to implement algorithms and perform experiments in the field of Machine Learning on Evolving Data Streams. scikit-multiflow is inspired in the popular frameworks scikit-learn, MOA and MEKA.

scikit-learn (Pedregosa et al., 2011) is the most popular open source software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forest, gradient boosting, k-means and DBSCAN, and is designed to inter-operate with the Python numerical and scientific packages NumPy and SciPy.

MOA (Bifet et al., 2010) is the most popular open source framework for data stream mining, with a very active growing community. It includes a collection of machine learning algorithms (classification, regression, clustering, outlier detection, concept drift detection and recommender systems) and tools for evaluation. Related to the WEKA project (Hall et al., 2009), MOA is also written in Java, while scaling to more demanding problems.

The MEKA project (Read et al., 2016) provides an open source implementation of methods for multi-label learning and evaluation. In multi-label classification, the aim is to predict multiple output variables for each input instance. This different from the 'standard' case (binary, or multi-class classification) which involves only a single target variable.

As a multi-output streaming framework, scikit-multiflow serves as a bridge between research communities that have flourished around the aforementioned popular frameworks, providing a common ground where they can thrive. scikit-multiflow assists on the democratization of Stream Learning by bringing this research field closer to the Machine Learning community, given the increasing popularity of the Python programing language. The objective is two-folded: First, fills the void in Python for a stream learning framework which can also interact with available tools such as scikit-learn and extends the set of available

Table 1: Available methods in scikit-multiflow. Methodologies on the left, and frameworks on the right of the vertical bar.

						Java		Python		
Algorithm	Classification	Regression	Single-Output	Multi-Output	Drift Detection	MOA	MEKA[†]	scikit-learn[†]	scikit-multiflow	**Reference**
kNN	✓		✓			✓	✓	✓	✓	Bishop (2006)
kNN + ADWIN	✓		✓			✓			✓	Bifet et al. (2018)
SAM kNN	✓		✓		✓	✓			✓	Losing et al. (2017)
Hoeffding Tree	✓		✓			✓			✓	Hulten et al. (2001)
Hoeffding Adaptive Tree	✓		✓		✓	✓			✓	Bifet et al. (2018)
FIMT-DD		✓	✓		✓	✓			✓	Bifet et al. (2018)
Adaptive Random Forest	✓		✓		✓	✓			✓	Gomes et al. (2017)
Oza Bagging	✓		✓		*	✓			✓	Oza (2005)
Leverage Bagging	✓		✓		✓	✓			✓	Bifet et al. (2018)
Multi-output Learner	✓	✓	✓	✓	*	✓	✓	✓	✓	Bishop (2006)
Classifier Chains	✓		✓	✓	*		✓	✓	✓	Read et al. (2016)
Regressor Chains		✓	✓	✓	*		✓	✓	✓	Read et al. (2016)
SGD	✓	✓	✓			✓	✓	✓	✓	Bishop (2006)
Naive Bayes	✓		✓			✓	✓	✓	✓	Bishop (2006)
MLP	✓	✓	✓				✓	✓	✓	Bishop (2006)
ADWIN					✓	✓			✓	Bifet et al. (2018)
DDM					✓	✓			✓	Gama et al. (2004)
EDDM					✓	✓			✓	Bifet et al. (2018)
Page Hinkley					✓	✓			✓	Page (1954)

[*] Depending on the base learner.
[†] We have only listed incremental methods for data-streams; MEKA and scikit-learn have many other batch-learning models available. MEKA in particular, has many problem-transformation methods which may be incremental depending on the base learner (it is able to use those from the MOA framework).

state-of-the-art methods on this platform. Second, provides a set of tools to facilitate the development of stream learning research, an example is (Montiel et al., 2018).

It is important to notice that scikit-multiflow complements scikit-learn, whose primary focus is batch learning, expanding the set of free and open source tools for Stream Learning. In addition, scikit-multiflow can be used within Jupyter Notebooks, a popular interface in the Data Science community. Special focus in the design of scikit-multiflow is to make it friendly to new users and familiar to experienced ones.

scikit-multiflow contains stream generators, learning methods, change detectors and evaluation methods. Stream generators include: Agrawal, Hyperplane, Led, Mixed, Random-RBF, Random-RBF with drift, Random Tree, SEA, SINE, SEA, STAGGER, Waveform, Multi-label, Regression and Concept-Drift. Available evaluators correspond to Prequential and Hold-Out evaluations, both supporting multiple performance metrics for *Classification* (Accuracy, Kappa Coefficient, Kappa T, Kappa M), *Multi-Output Classification* (Hamming Score, Hamming Loss, Exact Match, Jaccard Index) *Regression* (Mean Squared Error, Mean Absolute Error) and *Multi-Output Regression* (Average Mean Squared Error, Average Mean Absolute Error, Average Root Mean Squared Error). Learning methods and change detectors are listed in Table 1. This table also serves to outline the position of scikit-multiflow with respect to other open source frameworks.

2. Stream Data Mining Notation and Background

Consider a continuous stream of data $A = \{(\vec{x}_t, y_t)\} | t = 1, \ldots, T$ where $T \to \infty$. Input $\vec{x}_t$ is a feature vector and y_t the corresponding target where y is continuous in the case of regression and discrete for classification. The objective is to predict the target y for an unknown $\vec{x}$. In traditional single-output models, we deal with a single target variable for which one corresponding output is produced per test instance. Multi-output models can produce multiple outputs to assign to multiple target variables $\vec{y}$ for each test instance.

Different to batch learning, where all data (X, y) is available for training; in stream learning, training is performed incrementally as new data $(\vec{x}_i, y_i)$ is available. Performance P of a given model is measured according to some loss function that evaluates the difference between the set of expected labels Y and the predicted ones $\hat{Y}$. Hold-out evaluation is a popular performance evaluation method for batch and stream settings, where tests are performed in a separate test set. Prequential-evaluation (Dawid, 1984) or interleaved-test-then-train evaluation, is a popular performance evaluation method for the stream setting only, where tests are performed on new data before using it to train the model.

3. Architecture

The base class in scikit-multiflow is StreamModel which contains the following abstract methods to be implemented by its subclasses:

- fit — Trains a model in a batch fashion. Works as an interface to batch methods that implement a fit() function such as scikit-learn methods.
- partial_fit — Incrementally trains a stream model.
- predict — Predicts the target's value in supervised learning methods.
- predict_proba — Calculates per-class probabilities in classification problems.

A StreamModel object interacts with two other objects: a Stream object and (optionally) a StreamEvaluator object. The Stream object provides a continuous flow of data on request. The StreamEvaluator performs multiple tasks: queries the stream for data, trains and tests the model on the incoming data and continuously tracks the model's performance. The sequence to train a Stream Model and track its performance using prequential evaluation in scikit-multiflow is outlined in Figure 1.

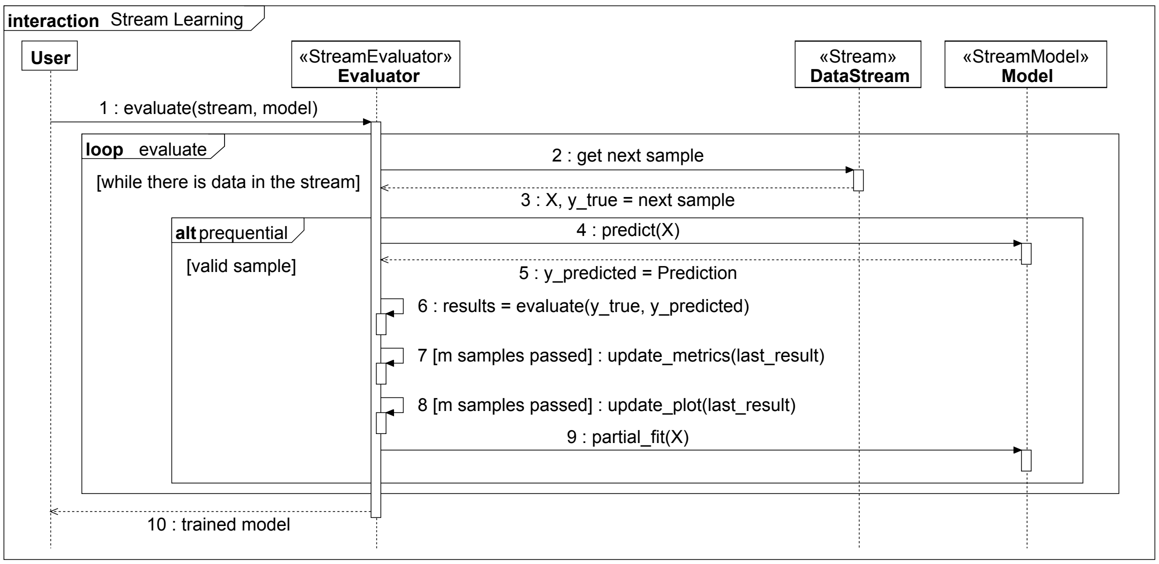

Figure 1: Training and testing a stream model using scikit-multiflow. This sequence corresponds to prequential evaluation.

4. Development

The scikit-multiflow package is distributed under the BSD License. Development follows the FOSS principles and encompasses:

- A webpage, `https://scikit-multiflow.github.io/`, including documentation and user guide. Both, documentation and user guide, are living documents that are continuously updated to reflect the current stage of scikit-multiflow.
- Version control via git. The source code of the package is publicly available on Github at `https://github.com/scikit-multiflow/scikit-multiflow`
- Package deployment and software quality are enforced via continuous integration and functional testing, `https://travis-ci.org/scikit-multiflow/scikit-multiflow`

References

Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. MOA: Massive online analysis. *Journal of Machine Learning Research*, 11(May):1601–1604, 2010.

Albert Bifet, Ricard Gavalda, Geoff Holmes, and Bernhard Pfahringer. *Machine Learning for Data Streams with Practical Examples in MOA*. MIT Press, 2018. `https://moa.cms.waikato.ac.nz/book/`.

Christopher M Bishop. Pattern recognition and machine learning. 2006.

A Philip Dawid. Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society. Series A (General)*, pages 278–292, 1984.

João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with Drift Detection. pages 286–295, 2004. ISSN 0302-9743. doi: 10.1007/978-3-540-28645-5_29.

Heitor M. Gomes, Albert Bifet, Jesse Read, Jean Paul Barddal, Fabrício Enembreck, Bernhard Pfharinger, Geoff Holmes, and Talel Abdessalem. Adaptive random forests for evolving data stream classification. *Machine Learning*, 106(9-10):1469–1495, 2017. ISSN 15730565. doi: 10.1007/s10994-017-5642-8.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11 (1):10–18, November 2009. ISSN 1931-0145. doi: 10.1145/1656274.1656278.

Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining time-changing data streams. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '01*, 18:97–106, 2001. ISSN 10844627. doi: 10.1145/502512.502529.

Viktor Losing, Barbara Hammer, and Heiko Wersing. KNN classifier with self adjusting memory for heterogeneous concept drift. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 1:291–300, 2017. ISSN 15504786. doi: 10.1109/ICDM.2016.141.

Jacob Montiel, Albert Bifet, Viktor Losing, Jesse Read, and Talel Abdessalem. Learning fast and slow: A unified batch/stream framework. In *Big Data (Big Data), 2018 IEEE International Conference on*. IEEE, 2018.

N.C. Oza. Online Bagging and Boosting. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2340–2345. IEEE, 2005. ISBN 0-7803-9298-1. doi: 10.1109/ICSMC.2005.1571498.

Ewan S Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. MEKA: A multi-label/multi-target extension to Weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016.