Journal of Machine Learning Research 22 (2021) 1-7 Submitted 7/21; Revised 12/21; Published 12/21

# Toolbox for Multimodal Learn (`scikit-multimodallearn`)

**Dominique Benielli** FIRSTNAME.LASTNAME[AT]UNIV-AMU.FR
**Cécile Capponi** FIRSTNAME.LASTNAME[AT]LIS-LAB.FR
**Sokol Koço** FIRSTNAME.LASTNAME[AT]MINES-STETIENNE.FR
**Hachem Kadri** FIRSTNAME.LASTNAME[AT]LIS-LAB.FR
**Riikka Huusari** FIRSTNAME.LASTNAME[AT]LIS-LAB.FR
*Department of Computer Science Aix-Marseille University, CNRS, LIS, 13013 Marseille, France*

**Baptiste Bauvin** FIRSTNAME.LASTNAME[AT]LIS-LAB.FR
**François Laviolette** FIRSTNAME.LASTNAME[AT]IFT.ULAVAL.CA
*Pavillon Adrien-Pouliot, Local PLT-3908, 1065, av. de la Médecine, Université Laval, Québec (QC) G1V 0A6, Canada*

**Editor:** Joaquin Vanschoren

## Abstract

`scikit-multimodallearn` is a Python library for multimodal supervised learning, licensed under Free BSD, and compatible with the well-known `scikit-learn` toolbox (Pedregosa et al., 2011). This paper details the content of the library, including a specific multimodal data formatting and classification and regression algorithms. Use cases and examples are also provided.

**Keywords:** multimodal, multiview, classifier, supervised learning, algorithms

## 1. Introduction

Learning from multiple views deals with the integration of different representations of the data, that can be either redundant, complementary, independent, or contradictory, in order to solve a learning problem (Cesa-Bianchi et al., 2010; Sun, 2013). Thus, learning over all the views is expected to produce a final model that performs better than models learnt on individual views separately. Multi-view learning is a setting of machine learning which was initially devoted to semi-supervised tasks with view agreement (Blum and Mitchell, 1998), and later approaches penalizing view disagreements (Janodet et al., 2009), while mainly considering only two views. Surfing along the supervised side of multi-view learning, the assumption on view agreement is no longer considered, for the focus is set on exploring the complementarities (or diversity) of views. Methods of supervised multi-view learning include Multiple Kernel Learning (Bach et al., 2004; Gönen and Alpaydın, 2011), multi-view machines (Cao et al., 2016), ensemble-based methods (Koço and Capponi, 2011; Goyal et al., 2017), etc.

This paper presents a collaborative toolbox which implements algorithms for multimodal supervised learning where each labelled example is observed under several views. Compatible with `scikit-learn` and `numpy` libraries, it comes with a intuitive input format for multi-view data sets which allows the use embedded algorithms in model selection facilities of `sklearn`. In this paper, we consider that mutli-view and multi-modal can be

used interchangeably to qualify the type of data set we present, even if it can have different meaning in some fields.

## 2. Toolbox `scikit-multimodallearn` at a Glance

The toolbox `scikit-multimodallearn` aims at providing a unified format for multi-view data representation by implementing a data structure in the form on a python class inheriting from the numpy array class. Empowered by this novel data structure, we propose the implementation of four multi-view algorithms as python classes each implementing a classifier and rigorously respecting the `scikit-learn` syntax for learning algorithms. A detailed documentation and a technical manual are available online.[1]

### 2.1 Installing and Discovering `scikit-multimodallearn`

The installation of the toolbox is made easy by the use of `pip`: one just has to execute `pip install scikit-multimodallearn` in a terminal, or `pip install -e .` if you have sources in local. If needed, the package can be downloaded from Github. [2] The development has been performed using continuous integration with Docker and automated tests, covering 90% of the code. A technical documentation, including the automated Sphinx documentation and examples are available online [3] to allow collaborative development.

### 2.2 Input Data Format

Multi-view data are encoded by `MultiModalArray`, a python class which encapsulates information in `numpy` array-like format adapted for multi-modal learning. A first attempt to deal with sparse data representation is also included in the `MultiModalSparseArray` class.

This groundwork is crucial for multi-view algorithm development as it allows to implement algorithms that can treat multi-view data sets while being compatible with the very useful `scikit-learn` library. In particular, algorithms implemented with `scikit-multimodallearn` take the exact same arguments as their `scikit-learn` basic counterparts: `fit`, `predict` and `score`, take either a `MultiModalArray`, `MultiModalSparseArray`, a `dictionary` or a simple `numpy_array` coupled with a list of view indices range in `views_ind` as input, enabling `scikit-multimodallearn` to be as versatile and user-friendly as possible. In the following example, a multiview data set is instantiated with a multi-view version of MNist (Deng, 2012) for which we generated the HOG (Triggs and Dalal, 2005) in 12 directions, and selected 3 random directions for each view.

```
>>> import numpy as np
>>> from multimodal.datasets.data_sample import MultiModalArray
>>> from multimodal.datasets.base import load_dict
>>> from multimodal.tests.data.get_dataset_path import get_dataset_path
>>> dic_digit = load_dict(get_dataset_path("multiview_mnist.npy"))
>>> y = np.load(get_dataset_path("mnist_y.npy"))
>>> XX = MultiModalArray(dic_digit)
```

1. Hosted here https://dev.pages.lis-lab.fr/scikit-multimodallearn/ .
2. Hosted here https://github.com/dbenielli/scikit-multimodallearn .
3. Hosted here http://dev.pages.lis-lab.fr/scikit-multimodallearn/tutorial/auto_examples/index.html .

```
>>> XX.shape
(5000, 768)
>>> XX.n_views
4
>>> XX.views_ind
array([  0,  192, 384, 576, 768])
```

### 2.3 Architecture: Classifier Class and Algorithms

As a second major contribution, this library provides four multi-view classification algorithms that are designed to learn with multiple views (i.e. not restricted to only two views). These algorithms fall into two categories: kernel-based binary classifiers such as MVML (Huusari et al., 2018), a classification version of the MKL algorithm (Kloft et al., 2011); and multi-class boosting algorithms such as MumBo (Koço and Capponi, 2011) and MuCombo (Koço, 2013):

- **MVML** uses machinery from the theory of operator-valued kernels and vector-valued RKHS to learn simultaneously a vector-valued function to solve the supervised learning problem, and a metric acting between each pair of views in the data.
- **lp-MKL** provides co-regularization and multiple kernel learning which are two well known kernel-based frameworks for learning in the presence of multiple views of data.
- **Mumbo** encourages the collaboration between major and minor views, in order to enhance the performances of classifiers usually learnt only on the major view.
- **MuComBo** is adapted to imbalanced learning problems, its adds a balanced weight on classes to the collaboration between view of Mumbo.

The algorithms are implemented in four main classes : `MVML`, `MKL`, `MumboClassifier`, `MuComboClassifier`, and two parent abstract meta classes `MKernel` and `UBoosting` for code factorization. As required by `scikit-learn`, the main class inherits from `BaseEstimator` or `BaseEnsemble` and `ClassifierMixin` from `sklearn.base`.

In addition, the kernel-based algorithms scale up to large training sets using a block-wise Nyström approximation of the multi-view kernel matrix (proposed as option).

Finally, as the code for these algorithms respects the guidelines given by `scikit-learn`, it is easy to add multi-view algorithms to the library using the tools provided by `scikit-learn` to test the compatibility and the data format provided by `scikit-multimodallearn`.

### 2.4 Multimodal Estimators Use Case

We present in the following a use case for initializing a `MumboClassifier` estimator.

```
>>> # import
>>> from multimodal.boosting.mumbo import MumboClassifier
>>> from sklearn.tree import DecisionTreeClassifier
>>> from sklearn.model_selection import train_test_split, cross_val_score
>>> # instantiation of  weak classifier
>>> base_estimator = DecisionTreeClassifier(max_depth=4)
>>> # split data in train and test
>>> X_train, X_test, y_train, y_test = train_test_split(XX, y)
```

```
>>> # instantiation of MumboClassifier classifier
>>> clf = MumboClassifier(base_estimator=base_estimator,
                          n_estimators=4, random_state=7)
>>> # call fit method
>>> clf.fit(X_train, y=y_train)
>>> # call predict method
>>> y_pred = clf.predict(X_test))
>>> # call score method
>>> clf.score(X_test, y_test)
0.9666666666666667
>>> # result
>>> np.mean(y_pred.ravel() == y_test.ravel()) * 100
   96.66666666666667
```

`MuComBoClassifier`, `MVML` and `MKL` can be used the same way, and `MuComBoClassifier` with `base_estimator` parameter initialized to a (list of) monoview classifier(s), the default classifier being the default `DecisionTreeClassifier` of `scikit-learn`.

## 3. `scikit-learn` Compatibility

A crucial advantage stemming from the compatibility of `scikit-multimodallearn` with `scikit-learn` is the underlying inheritance of several functionalities, such as :

- *train_test_split* of `scikit-learn` can be used with `MultiModalArray` as demontrated in 2.4;
- `cross_val_score` can be used transparently on multimodal estimator for multiviews data.

```
>>> # Instantiating MuComboClassifier
>>> clfm = MuComboClassifier(base_estimator=base_estimator)
>>> from sklearn.model_selection import cross_val_score
>>> # usage cross_val_score on MuComboClassifier
>>> cross_val_score(clfm, XX, y, cv=5)
array([0.96666667, 0.96666667, 0.9, 0.93333333, 1. ])
```

- `grid_search`, can also be used directly, for example, through `OneVsOneClassifier` for the non-multiclass MVML.

```
>>> # import section
>>> from sklearn.multiclass import OneVsOneClassifier
>>> from sklearn.model_selection import GridSearchCV
>>> from multimodal.kernels.mvml import MVML
>>> # Instantiating a one versus one classifier based on MVML
>>> est = OneVsOneClassifier(MVML(lmbda=0.1, eta=1, nystrom_param=0.2))
>>> # Defining the hyper-parameter grid
>>> param = {'estimator__learn_A': (1, 3), 'estimator__learn_w': (0, 1),
 'estimator__n_loops': (6, 10), 'estimator__nystrom_param': (1.0, 0.3),
 'estimator__kernel': ('linear', 'polynomial') ,
 'estimator__lmbda': (0.1,), 'estimator__eta': (1,)}
>>> # Instantiating the grid search object and fitting it
```

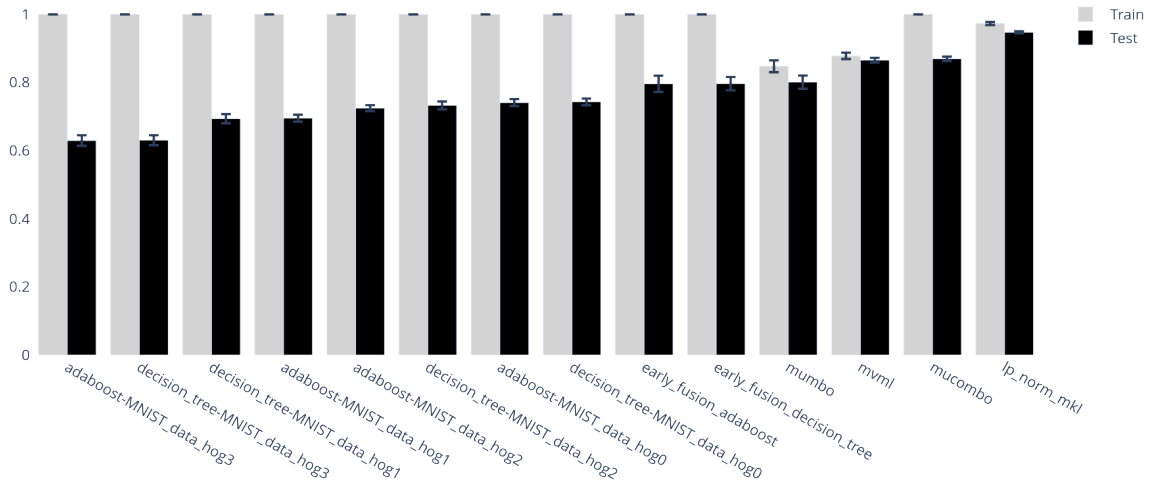


Figure 1: Accuracy results on a multi-view version of the multi-class MNist data set, computed with SuMMIT (Bauvin et al., 2021) in which early and late fusion are already implemented.

```
>>> grid = GridSearchCV(est, param, cv = 5).fit(XX, y)
>>> # Getting the best score
>>> grid.best_score_
0.9693934335002783
>>> grid.best_params_
{'estimator__eta': 1, ... 'estimator__n_loops': 6, 'estimator__nystrom_param
    ': 1.0}
```

## 4. Results

Figure 1 shows the results of a benchmark, with 5-folds cross validation on the multi-view version of MNist presented in Section 2.2. We tested a mono-view decision tree and Adaboost (Schapire, 2013) on each view, their early fusion versions and the four algorithms of `scikit-multimodallearn`. All the implemented algorithms output higher accuracy scores than the mono-view approaches and the naive fusion methods. MuCombo, despite being dedicated to imbalance problems, still displays an interesting score for this balanced task.

## 5. Conclusion

`scikit-multimodallearn` offers specialized algorithms dedicated to multimodal supervised learning developed in Python3 on free licence, with an easy and transparent use for the user accustomed to the uses of `scikit-learn`.

We are currently working on a more robust integration of sparse matrices and the addition of a data format allowing to load the views dynamically when needed, such as

HDF5. Finally, the `scikit-learn` and SuMMIT compatibility will allow the multi-view community to develop their own algorithms on this framework to build a large library.

## Acknowledgements

This work is supported by National Science and Engineering Research Council of Canada (NSERC) Discovery grant 262067, and granted by Lives Project (ANR-15-CE23-0026).

## References

Francis Bach, Gert Lanckriet, and Michael Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In International Conference on Machine Learning (ICML), pages 41–48, 2004.

Baptiste Bauvin, Dominique Benielli, Cécile Capponi, and François Laviolette. Integrating and reporting full multi-view supervised learning experiments using SuMMIT. working paper or preprint, April 2021. URL `https://hal.archives-ouvertes.fr/hal-03197079`.

Avrim B. Blum and Tom M. Mitchell. Combining labeled and unlabeled data with co-training. In 11th Annual Conference on Computational Learning Theory (COLT), pages 92–100, 1998.

Bokai Cao, Hucheng Zhou, Guoqiang Li, and Philip S Yu. Multi-view machines. In Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, pages 427–436, 2016.

Nicolo Cesa-Bianchi, David R. Hardoon, and Gayle Leen. Guest Editorial: Learning from multiple sources. Machine Learning, 79(1-2):1–3, 2010.

Li Deng. The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine, 29(6):141–142, 2012.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. The Journal of Machine Learning Research, 12:2211–2268, 2011.

Anil Goyal, Emilie Morvant, Pascal Germain, and Massih-Reza Amini. Pac-bayesian analysis for a two-step hierarchical multiview learning approach. In Michelangelo Ceci, Jaakko Hollmén, Ljupčo Todorovski, Celine Vens, and Sašo Džeroski, editors, Machine Learning and Knowledge Discovery in Databases, pages 205–221, Cham, 2017. Springer International Publishing. ISBN 978-3-319-71246-8.

Riikka Huusari, Hachem Kadri, and Cécile Capponi. Multi-view metric learning in vector-valued kernel spaces. In Amos Storkey and Fernando Perez-Cruz, editors, Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics, volume 84 of Proceedings of Machine Learning Research, pages 415–424, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR. URL `http://proceedings.mlr.press/v84/huusari18a.html`.

Jean-Christophe Janodet, Marc Sebban, and Henri-Maxime Suchier. Boosting Classifiers built from Different Subsets of Features. Fundamenta Informaticae, 94(2009):1–21, July 2009. doi: 10.3233/FI-2009-131. URL https://hal.archives-ouvertes.fr/hal-00403242.

Marius Kloft, Ulf Brefeld, Sören Sonnenburg, and Alexander Zien. lp-norm multiple kernel learning. Journal of Machine Learning Research, 12(26):953–997, 2011. URL http://jmlr.org/papers/v12/kloft11a.html.

Sokol Koço. "Tackling the uneven views problem with cooperation based ensemble learning methods. PhD thesis, Aix-Marseille Université, 2013.

Sokol Koço and Cécile Capponi. A boosting approach to multiview classification with cooperation. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II, pages 209–228, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-23783-6. URL https://link.springer.com/chapter/10.1007/978-3-642-23783-6_14.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

Robert E Schapire. Explaining adaboost. In Empirical inference, pages 37–52. Springer, 2013.

Shiliang Sun. A survey of multi-view machine learning. Neural Computing and Applications, 23(7):2031–2038, Dec 2013. ISSN 1433-3058. doi: 10.1007/s00521-013-1362-6. URL https://doi.org/10.1007/s00521-013-1362-6.

Bill Triggs and Navneet Dalal. Histograms of oriented gradients for human detection. In 2013 IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 886–893, Los Alamitos, CA, USA, jun 2005. IEEE Computer Society. doi: 10.1109/CVPR.2005.177. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2005.177.