Towards Meta-learning over Data Streams (Abstract)

Jan N. van Rijn[1] and **Geoffrey Holmes**[2] and **Bernhard Pfahringer**[3] and **Joaquin Vanschoren**[4]

Modern society produces vast streams of data. Many stream mining algorithms have been developed to capture general trends in these streams, and make predictions for future observations, but relatively little is known about which algorithms perform particularly well on which kinds of data. Moreover, it is possible that the characteristics of the data change over time, and thus that a different algorithm should be recommended at various points in time. Figure 1 illustrates this. As such, we are dealing with the Algorithm Selection Problem [9] in a data stream setting. Based on measurable *meta-features* from a window of observations from a data stream, a *meta-algorithm* is built that predicts the best classifier for the next window. Our results show that this meta-algorithm is competitive with state-of-the-art data streaming ensembles, such as `OzaBag` [6], `OzaBoost` [6] and `Leveraged Bagging` [3].

We first construct a meta-dataset consisting of 49 data streams, generated using various data stream generators from the MOA workbench [2], including the Rotating Hyperplane Generator and Random RBF Generator. In addition, we use a newly created Bayesian Network Generator, which takes a dataset as input, preferably consisting of real-world data and a reasonable amount of features, and builds a Bayesian Network using this dataset as input [12]. The Bayesian Network is then used to generate a data stream, determining each feature of each instance using the probability tables. These streams all contain 1,000,000 instances. We also include commonly used large datasets, such as Covertype, Pokerhand and the 20 Newsgroups dataset.

We run three types of classifiers over these datasets [8]. These are instance incremental classifiers, which learn from each example as it arrives, batch incremental classifiers, which learn from batches of examples, and ensembles of classifiers. The score of these classifiers is recorded at each window of 1,000 instances. Furthermore, we calculate various meta-features for all of these intervals, most of which are described in [10]. These meta-features are typically categorised as one of the following: simple (number of instances, number of attributes, number of classes), statistical (mean standard deviation of attributes, mean kurtosis of attributes, mean skewness of attributes), information theoretic (class entropy, mean entropy of attributes, noise-signal ratio) or landmarkers [7] (performance of a simple classifier on the data). We also introduce stream-specific meta-features based on change detection, which count the number of changes detected by the `ADWIN` [1] and `DDM` [4] change detectors.

The results of all experiments, as well as the generated datasets, classifiers used, and the meta-dataset itself, are available on OpenML [11].

[1] Leiden University, Leiden, Netherlands, j.n.van.rijn@liacs.leidenuniv.nl
[2] University of Waikato, Hamilton, New Zealand, geoff@cs.waikato.ac.nz
[3] University of Waikato, Hamilton, New Zealand,bernhard@cs.waikato.ac.nz
[4] Eindhoven University of Technology, Eindhoven, Netherlands, j.vanschoren@tue.nl

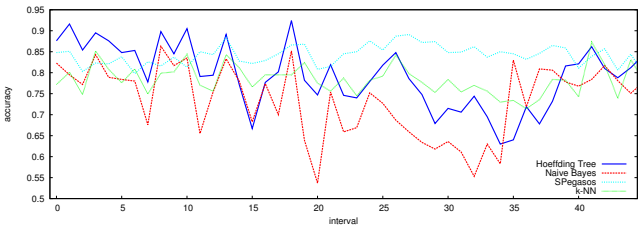

Figure 1: Performance of four instance incremental classifiers on intervals of the electricity dataset. Each interval contains 1,000 instances.

We now aim to determine whether this meta-knowledge can improve the predictive performance of data stream algorithms. We run a sliding window of 1,000 examples over each of the base data streams, and train a meta-algorithm using the meta-features and classifier scores for that window to predict which classifier should be used in the next window. The meta-algorithm is a `Random Forest` using 100 trees and 10 attributes, as implemented in Weka [5]. We distinguish between *meta-level accuracy* and *base-level accuracy*. Meta-level accuracy indicates how the meta-algorithm performs on the meta-learning task of predicting the best algorithm for a given window; base-level accuracy indicates how an ensemble of these base classifiers would actually perform on the base data stream, using the meta-algorithm to decide which base classifier to use for each window. The choice of meta-algorithm and the window size were determined experimentally.

Table 1 shows the results obtained from this experiment. We evaluate how well the meta-learning selects between 13 base stream classifiers, listed in Table 2. All classifiers are run with the default parameter settings as selected in MOA [2]. As described above, we can distinguish between three different types of stream mining algorithms, and we evaluate how the meta-learning approach performs within these subgroups as well.

Column A indicates the number of classifiers of each type, also indicated in Table 2. Column "Majority" denotes which classifier is the overall best in each group; here `HT` is short for `Hoeffding Trees`, `SMO` stands for a `Support Vector Machine` with a `Polynomial Kernel` and `LB-HT` means `Leveraged Bagging Hoeffding Trees`. The column "Percentage" shows the percentage of 1,000-example windows where this overall best algorithm wins. Since the meta-learner has to predict which base classifier to use in each window, this value represents the default accuracy of the meta-learning task.

Next, $\texttt{RF}_{meta}$ shows the accuracy of the `Random Forest` meta-

Table 1: Results of algorithm selection in the stream setting.

Task	A	Majority	Percentage	$\texttt{RF}_{meta}$	$\texttt{ZeroR}_{base}$	$\texttt{RF}_{base}$	$\texttt{MAX}_{base}$
Instance incremental	5	HT	59.75	80.78	80.98	84.07	84.59
Batch incremental	4	SMO	65.56	68.17	74.38	75.33	76.02
Ensembles	4	LB-HT	57.78	56.20	84.27	85.15	86.12
All classifiers	13	LB-HT	50.97	50.92	84.27	85.31	86.30

Table 2: Algorithms used in the experiments.

Key	Classifier	Type	Parameters
NB	NaiveBayes	Instance incremental	
SGD	Stochastic Gradient Descent	Instance incremental	
SPeg	SPegasus	Instance incremental	
k-NN	k Nearest Neighbour	Instance incremental	$k = 10, w = 1000$
HT	Hoeffding Tree	Instance incremental	
SMO	Support Vector Machine / Polynomial Kernel	Batch incremental	$w = 1000$
J48	C4.5 Decision Tree	Batch incremental	$w = 1000$
REP	Reduced-Error Pruning Decision Tree	Batch incremental	$w = 1000$
OneR	One Rule	Batch incremental	$w = 1000$
LB-kNN	Leveraging Bagging / k-NN	Ensemble	$k = 10, n = 10, w = 1000$
LB-HT	Leveraging Bagging / Hoeffding Tree	Ensemble	$n = 10$
Bag-HT	OzaBag / Hoeffding Tree	Ensemble	$n = 10$
Boost-HT	OzaBoost / Hoeffding Tree	Ensemble	$n = 10$

classifier in predicting the best classifier for a given window. The last three columns show the accuracy that can be obtained on the base data stream using three different strategies. Column $\texttt{ZeroR}_{base}$ shows the accuracy obtained by always selecting the best overall base classifier. For instance, the value in the "Ensembles" row shows the accuracy of an ensemble of Leveraged Bagged Hoeffding Trees, averaged over all data streams. $\texttt{RF}_{base}$ shows the accuracy obtained when the Random Forest meta-classifier predicts the base classifier to be used in each window, again averaged over all data streams. Finally, column $\texttt{MAX}_{base}$ shows the accuracy obtained if the meta-classifier always correctly predicted the best classifier for each window. Intuitively, $\texttt{RF}_{base}$ shows the performance of the meta-classifier, $\texttt{ZeroR}_{base}$ can be used as a baseline, and $\texttt{MAX}_{base}$ shows the maximum score that the meta-classifier could have obtained.

Determining the best instance incremental classifier yields good results. In more than 80% of the cases, the correct classifier is predicted. This also translates into good base-level performance. An ensemble of our meta-classifier and only the 5 instance incremental classifiers, which is markedly cheaper to train, yields a score of 84.07%, which not only outperforms the best overall instance incremental classifier, a Hoeffding Tree with 80.98% accuracy, but is also comparable to the best overall base classifier, a Leveraged Bagged Hoeffding Trees ensemble (with 10 base-classifiers), which scores 84.27%. Moreover, it also outperforms the other ensembles, OzaBag (82.58%) and OzaBoost (80.55%). The Random Forest meta-learner has more difficulty selecting among all 13 base-classifiers, which shows room for progress, but even then it performs slightly better than the overall best base classifier. Furthermore, the $\texttt{RF}_{base}$ performances are in many cases close to the maximal possible value, $\texttt{MAX}_{base}$. This indicates that the main challenge is to find ways to improve this limit. Better results are likely to be obtained using parameter optimisation, and by using a larger set of algorithms.

REFERENCES

[1] A. Bifet and R. Gavalda. Learning from Time-Changing Data with Adaptive Windowing. In *SDM*, volume 7, pages 139–148. SIAM, 2007.

[2] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer. MOA: Massive Online Analysis. *J. Mach. Learn. Res.*, 11:1601–1604, 2010.

[3] A. Bifet, G. Holmes, and B. Pfahringer. Leveraging Bagging for Evolving Data Streams. In *Machine Learning and Knowledge Discovery in Databases*, volume 6321 of *Lecture Notes in Computer Science*, pages 135–150. Springer, 2010.

[4] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with Drift Detection. In *SBIA Brazilian Symposium on Artificial Intelligence*, volume 3171 of *Lecture Notes in Computer Science*, pages 286–295. Springer, 2004.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA Data Mining Software: An Update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[6] Nikunj C Oza. Online Bagging and Boosting. In *Systems, man and cybernetics, 2005 IEEE international conference on*, volume 3, pages 2340–2345. IEEE, 2005.

[7] Bernhard Pfahringer, Hilan Bensusan, and Christophe Giraud-Carrier. Tell me who can learn you and I can tell you who you are: Landmarking various learning algorithms. In *Proceedings of the 17th international conference on machine learning*, pages 743–750, 2000.

[8] J. Read, A. Bifet, B. Pfahringer, and G. Holmes. Batch-Incremental versus Instance-Incremental Learning in Dynamic and Evolving Data. In *Advances in Intelligent Data Analysis XI*, pages 313–323. Springer, 2012.

[9] J. R. Rice. The Algorithm Selection Problem. *Advances in Computers*, 15:65118, 1976.

[10] Q. Sun and B. Pfahringer. Pairwise meta-rules for better meta-learning-based algorithm ranking. *Machine learning*, 93(1):141–161, 2013.

[11] J. N. van Rijn, B. Bischl, L. Torgo, B. Gao, V. Umaashankar, S. Fischer, P. Winter, B. Wiswedel, M. R. Berthold, and J. Vanschoren. OpenML: A Collaborative Science Platform. In *Machine Learning and Knowledge Discovery in Databases*, pages 645–649. Springer, 2013.

[12] J. N. van Rijn, G. Holmes, B. Pfahringer, and J. Vanschoren. The Bayesian Network Generator: A data stream generator. Technical Report 03/2014, Computer Science Department, University of Waikato, 2014.