A META-LEARNING METHOD FOR CONCEPT DRIFT

Runxin Wang, Lei Shi, Mícheál Ó. Foghlú and Eric Robson
Telecommunications Software & Systems Group Waterford Institute of Technology, Waterford, Ireland

Keywords: Data Mining, Supervised Learning, Concept Drift, Meta-Learning, Evolving Data.

Abstract: The knowledge hidden in evolving data may change with time, this issue is known as concept drift. It often causes a learning system to decrease its prediction accuracy. Most existing techniques apply *ensemble methods* to improve learning performance on concept drift. In this paper, we propose a novel meta learning approach for this issue and develop a method: Multi-Step Learning (MSL). In our method, a MSL learner is structured in a recursive manner, which contains all the base learners maintained in a hierarchy, ensuring the learned concepts are traceable. We evaluated MSL and two ensemble techniques on three synthetic datasets, which contain a number of drastic concept drifts. The experimental results show that the proposed method generally performs better than the ensemble techniques in terms of prediction accuracy.

1 INTRODUCTION

The Machine Learning (ML) and Data Mining (DM) communities have for many years been concerned with the problem of *concept drift*. Essentially if a learning model is trained with one set of training data, the risk is that over time this training set becomes less relevant to the new data being analysed, and thus the new concepts drift away from those previously learned. One practical example is weather prediction, where the prediction rules vary depending on the season (Widmer and Kubat, 1996). A survey of concept drift is provided by Tsymbal (Tsymbal, 2004). Most traditional ML algorithms are subject to concept drift where prediction accuracy decreases over time. These algorithms are often called *batch algorithms* as they are good at learning knowledge from data stored in batch, but become inefficient if the data grow dynamically and are exposed to concept drift.

Figure 1 gives an illustration of the concept drift problem in classification, where the concept has drifted from the first circle (old dataset) to the second (new dataset). Although classifier C_1 correctly classifies the data points in the old dataset; if C_1 is applied to the new dataset instead of C_2, it can be easily seen that some of the new data points would be misclassified. As shown, C_2 is the appropriate classifier to the new dataset.

One traditional method for dealing with evolving data is to relearn the knowledge based upon a *combined dataset* that consists of both the old data and the new data. However, this method is subject to "catastrophic forgetting" (Polikar et al., 2001) which refers to the issue of learning the new knowledge but forgetting the older knowledge. Therefore when presented with data relating to the original concepts, accurate prediction may no longer occur. In fact, concept drift requires a ML solution that does not just have one initial training phase, but has multiple phases of training, or some form of continuous training (e.g. incremental learning).

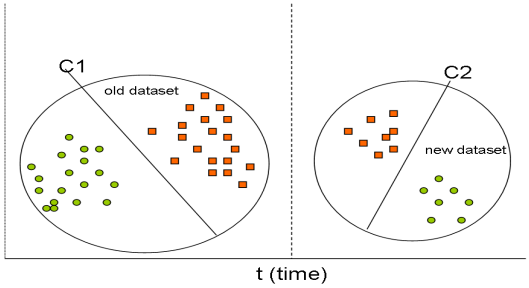

Figure 1: Two datasets are generated in different time blocks. The square examples and cycle examples respectively belong to different classes.

Some *ensemble techniques* have been proposed to address concept drift. The approach of these techniques is that the learning systems train multiple learners in different time windows. The periods (windows) observe a number of instances and the

Wang R., Shi L., Foghlú M. and Robson E..
A META-LEARNING METHOD FOR CONCEPT DRIFT.
DOI: 10.5220/0003095502570262
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2010), pages 257-262
ISBN: 978-989-8425-28-7

overall system combines these separate learners together into an integrated model using majority voting or weighted majority voting. These approaches are mainly inspired by Boosting (Freund and Schapire, 1997) and Bagging (Breiman, 1996).

In this paper we apply a meta-learning approach instead of either *combined dataset* or *ensemble techniques* to deal with concept drift. Our research focuses on Supervised Learning (SL), where we implement *Multi-Steps Learning (MSL)* to enhance SL performance on the data with concept drift. Nearly all of the methods dealing with concept drift can be seen as attempts to improve the performance of a Naïve Bayesian learning algorithm. These algorithms are good baseline for experimental analysis of a new method, as their accuracies are easy to adjust and therefore allow room to improve. In experiments, we compare *MSL* to a Bayesian *combined dataset* method and two Bayesian *ensemble techniques*.

The remainder of this paper is organized as follows. In Section 2, we discuss the properties of ensemble techniques, then briefly review some existing methods proposed to cope with concept drift. Section 3 presents our proposed method, *MSL*, for enhancing supervised learning on evolving data. Section 4 describes the experimental setup and presents the performance evaluation. Finally in Section 5 the key findings are emphasized, and some proposed future work is outlined.

2 RELATED WORK

Boosting and Bagging are the two most famous ensemble methods in ML and DM; they inspire solutions to many learning problems that require ensemble models, such as on-line learning, distributed learning, and incremental learning. They mainly provide two components: a mechanism of utilising instances in the available training sets, and a mechanism of combining the base learners. There are many existing combining rules (Kittler et al., 1998) in the field, where majority voting (used by Bagging) and weighted majority voting (used by Boosting) are the two mechanisms used most widely.

Streaming Ensemble Algorithm (SEA) (Street and Kim, 2001), is a pioneering method for dealing with concept drift in streaming data. It maintains a constant number of classifiers in its ensemble pool and when a new dataset is available, it performs majority classification on the new instances. It then re-evaluates the composite classifiers according to their classification accuracies and replaces some classifiers with new classifiers, if they are evaluated as under performing. The overall accuracy is improved by using the updated classifiers.

Beyond SEA, Bifet designed a new streaming ensemble method (Bifet et al., 2009), which enhances supervised learning by using two adapted versions of Bagging algorithms: adaptive windowing (ADWIN) and Adaptive-Size Hoeffding Tree (ASHT). The ADWIN is a change detector and the ASHT is an incremental anytime decision tree induction algorithm.

Dynamic weighted majority (DWM) (Kolter and Maloof, 2007) is a representative method using weighted majority voting. It maintains a couple of learners trained in different datasets in different time periods where each learner has a weight to specify how reliable it is. All the weights are updated over time according to the evaluation of the new datasets and the learners with low weights are removed or replaced with new learners. The overall system makes predictions using weighted majority voting among the base learners. One advantage of DWM is the number of base learners that should be used is initially specified and this set of base learners is continuously updated by the training process to reflect this.

There are also other methods for concept drift, which do not use ensemble techniques. Widmer (Widmer, 1997) was the first to propose tracking the context changes through meta-learning. Bach (Bach and Maloof, 2008) proposed to use only two learners (paired learners) to cope with all presented concepts in datasets.

3 MSL LEARNER

The implementation of our learning system, *Multi-Steps Learning (MSL)*, is presented in Algorithm 1. It consists of three types of learners, "old learners" that learn previous knowledge, "new learners" that learn current knowledge and "meta learners" that learn the experiences on how to select learners. Generally, if concept drift occurs, then a MSL learner will be composed by three learners, a new learner, a meta learner and an old learner. An important point to note is that the old learner could be also a MSL learner. In other words, although the new learner and the meta learner must be single learners, the old learner could actually be a composite learner. Figure 2 presents the structure of a MSL learner, which includes another MSL learner as its component (older learner). By using this structure, MSL can encode all the learned concepts in a traceable hierarchy. We will see this in the following section.

Zero or Singular Concept Drift in datasets. Initially MSL system builds a learner on the first dataset. If

no (zero) concept drift occurs, the MSL system will keep using that learner. However if the MSL system does discover concept drift in a new dataset, it will then save the existing learner as the "old learner" and subsequently train a "new learner" on the "hard set" (line 15 & 13). This groups the examples that the current learner fails to predict (line 10). Meanwhile, MSL generates the experience by labeling the examples with "old" if the MSL learner predicts correctly otherwise with "new" (line 11 & 12), and uses this to build a meta learner based on this experience (line 14). When provided with instances to analyse, the meta learner receives the instances at first and selects an appropriate learner between the new and the old learner to perform the final prediction. Equation 1 shows how a MSL learner produces its hypothesis on a given example X. A meta learner can be viewed as the public interface of a MSL learner, as it is always the first component to receive examples.

$$MSL(X) = \begin{cases} C_{old}(X) & \text{if } C_{meta}(X) = old \\ C_{new}(X) & \text{if } C_{meta}(X) = new \end{cases} \quad (1)$$

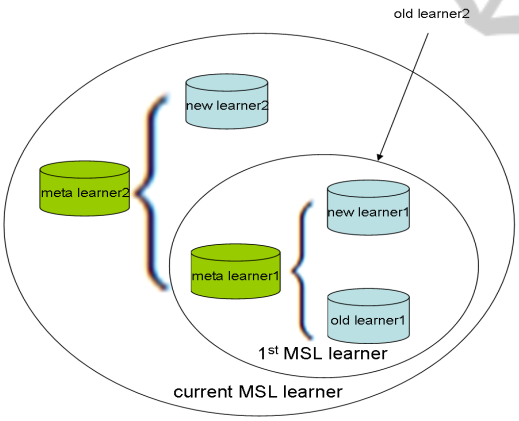

Figure 2: The structure of an overlap MSL learner. The old learner of the current MSL learner is pointing to a previous MSL learner.

Multiple Concept Drifts may appear in practice when more than one concept appears within the evolving data and some of the existing concepts may *recur* (Widmer and Kubat, 1996) during the lifetime of the data. To cope with multiple concepts, MSL recursively replaces an old learner with a current MSL learner (line 15) and accordingly trains a new learner (line 13) and a meta learner (line 14). Therefore every new concept can be learned, while all the previous concepts are preserved. Theoretically, each concept drift produces two learners, a "new learner" and a "meta learner", thus the number of base learners generated in MSL is a function of the number of concepts existing in the data:

$$|learners| = 2 \times (|concepts| - 1) + 1. \quad (2)$$

Algorithm 1: MSL Algorithm.

- **Input:** a sequence of examples.
- **Output:** a MSL learner.

1: **while** examples are continuously arriving **do**
2: **for** every t time **do**
3: Create a dataset d_t collecting the examples within t.
4: **end for**
5: **if** MSL learner = null **then**
6: Build MSL learner on d_t.
7: **else**
8: Evaluate MSL learner on d_t
9: **if** evaluation not pass **then**
10: Add all mispredicted examples into d_{hard}.
11: Label the mispredicted examples with o otherwise with n.
12: Add the labeled examples into d_{meta}.
13: Build learner C_{new} on d_{hard}.
14: Build learner C_{meta} on d_{meta}.
15: Set C_{old} equal to the current MSL learner.
16: Create a new MSL learner:

$$MSL(X) = \begin{cases} C_{old}(X) & \text{if } C_{meta}(X) = o \\ C_{new}(X) & \text{if } C_{meta}(X) = n \end{cases}$$

17: **end if**
18: **end if**
19: **end while**

Due to the fact that base learners can only learn on batch data, the MSL system creates a dataset for every t time to allow the learners to work on bounded datasets (line 3).

3.1 Complexity Analysis

The running time of MSL depends on the capability of the base learners and the input distribution. Let $f(n)$ be the training time required by a base learner to train n examples and $g(n)$ be the running time for predicting n examples. With one concept changing, in which case two learners are produced, then the running time of this case is $O(2f(n)+g(n))$. The running time of MSL on m datasets is $O(f(n)+k((n)+g(n))$ where $k \in [0,m]$ and $f(n)$ is the training time on the first dataset. Suppose that the number of datasets with changing concepts on m datasets has a Bernoulli distribution $\binom{m}{i} p^i (1-p)^{m-i}$ with parameter p, we could compute the average running time of MSL on m datasets as

$$E[O] = O(f(n) + \sum_{i=0}^{m} \binom{m}{i} p^i (1-p)^{m-i} \cdot i \cdot (2f(n) + g(n))) = O(f(n) + mp(2f(n) + g(n))).$$

4 EXPERIMENTS AND RESULTS

In our experiments we re-factored the Naïve Bayes in *weka* (Hall et al., 2009) and used it as the base learner. For the datasets, we used three synthetic datasets which are widely used in the literature, a detailed description of each dataset is given in the following sections. In the experiments, we mainly evaluated our MSL method and two ensemble techniques, Streaming Ensemble Algorithm (SEA) and Dynamic Weighted Majority (DWM). To ensure that the datasets were subject to concept drift, we also evaluated the *single learner* (a learner built on a singular dataset) and the *combined dataset* learner (a learner built on the dataset combining all the available training sets) to see if their prediction accuracies actually decrease as concepts change.

The general setup for the three type of datasets is presented as follows: for each type of dataset, we generated 5,500 instances with 10% noise, where 500 instances are reserved as a test set. The 5,000 instances are collected into 10 training sets, where 5 concepts appear in total. In other words, after the first dataset is generated, the remaining datasets experience concept drift 4 times. At the beginning of each series of experiments, we made a new benchmark of the concepts to check how dramatically the concepts change in each dataset.

4.1 SEA Data

SEA Data (Streaming Ensemble Algorithm) is an artificial dataset that simulates the environment of concept drift, it was originally introduced by Street and Kim (Street and Kim, 2001). This data consists of 3 attributes, one of which is irrelevant, and the other 2 attributes together define a concept in the following way: $x_1 + x_2 \leqslant v$, where v is a user-defined value. If v changes, the concept changes. As shown in Figure 3, the target concept changes suddenly when there are 1000 instances available. It can be seen that the *single learner* has a jump in error, while the MSL error does not increase as dramatically. It can be also seen from both Figure 3 and Figure 4 that, all the techniques' error rates are continuously increasing until the fifth concept arrives, where all the techniques made their worst performance. Afterwards, nearly all techniques achieved error reduction, where MSL got the greatest reduction. Generally, MSL and SEA have similar performance on this problem in terms of accuracy, while DWM seems to be slightly inferior to the others. One problem we discovered in MSL but not presented in the diagrams is that during the experiments MSL created more learners than it was expected. Recall equation 2, for data with 5 concepts it should of produced 9 learners, yet in fact we got 13 ± 2 learners in our experiments. We attribute this as a consequence of the high standard we used in evaluation, where MSL tends to believe a new concept arrived and accordingly train a new learner.

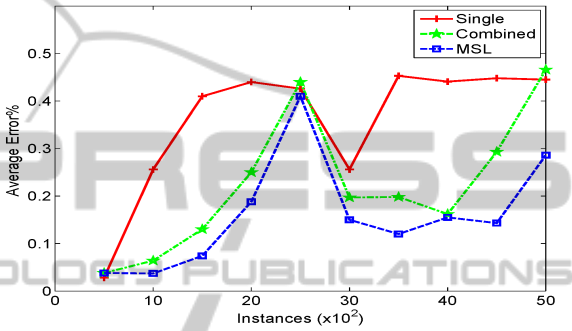

Figure 3: Experiments on SEA data.

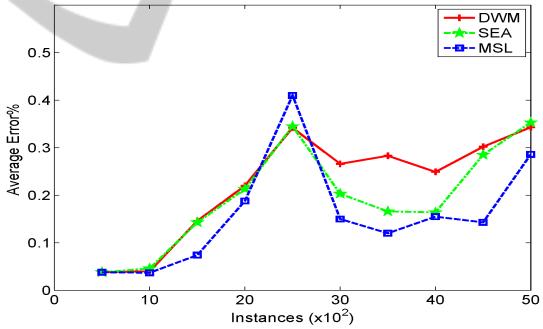

Figure 4: Experiments on SEA data.

4.2 Stagger Data

Stagger Data is another commonly used dataset for evaluating a learner's performance on concept drift. It was first created by Schlimmer and Granger (Schlimmer and Granger, 1986). Stagger data consists of three nominal attributes, color, shape and size. A concept is defined by a combination of the attribute values. For example, an instance of $color = red \oplus shape = circle \oplus size = small$ belongs to a target concept. The benchmark on figure 5 proves that the concept drifts do effect the performance of the techniques involved. On this problem, MSL achieved its best performance in terms of accuracy. As shown in Figure 6, comparing MSL to DWM and SEA, one can

see MSL outperforms the others nearly in every test block. Since a new concept is introduced at the test block when 1500 instances are available, both DWM and SEA continuously increased their error rates until they got the highest errors. MSL also increases its error rate from the same test block, but it almost manages to keep the error rate below 0.2 at every test block. With this dataset, the lower error rates suggest that MSL is capable of differentiating between the learned concepts and accordingly selected the most appropriate learner to perform the prediction. When only two concepts exist (1500 instances), the MSL learner made a much better performance than the others. However, as more concepts showed up, the difference between concepts become more difficult to identify, thus MSL's error rates increased as well. In this experiment, there were a couple of trials where we achieved the expected number of base learners in MSL, yet there were still some trials we did not achieve the expected number.

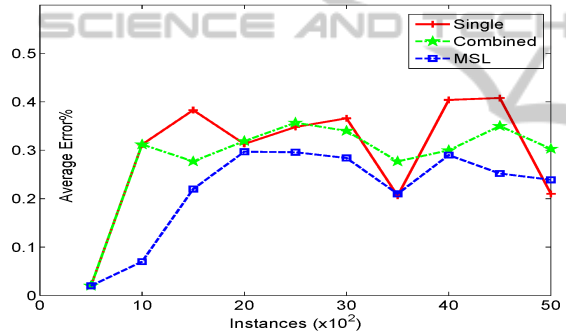

Figure 5: Experiments on Stagger data.

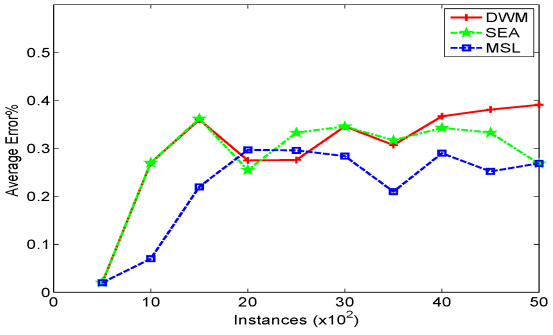

Figure 6: Experiments on Stagger data.

4.3 Moving Hyperplane Data

Moving Hyperplane Data was originally introduced by Hulten to test his proposed method VFDT (Hulten et al., 2001), an incremental decision tree learning algorithm. The concept is defined by the following equation:

$$\sum w_i x_i \leq v$$

where w_i is the weight of the i^{th} attribute, and x_i is the corresponding value. A concept is changed by moving the hyperplane (weights) of the equation. Figure 7 shows that though the error rates of the single learner and *combined dataset* learner vary largely, MSL works stably. In fact, as seen in Figure 8, the ensemble techniques also have a stable and good performance on this problem in terms of accuracy. With this dataset, DWM presented the best performance over all other algorithms, as shown on Figure 8. One interesting point shown in Figure 8 is that, there is a big gap showing on the point where 3000 instances are available and MSL has an error rate of 0.103, yet DWM and SEA only have 0.056 and 0.052 respectively. We examined the related trials and found that the fifth concept appeared at this point. We hypothesis far more concepts with insufficient instances made MSL unable to effectively classify the concepts. Indeed, MSL immediately recovers its accuracy at the follow point where more training data are available, which we believe it is supportive to our hypothesis. Again, we counted the learners that MSL produced for this problem, and we found that there are 13 ± 2 learners generated by MSL. This fact does not match with what we expected. We initially thought this issue would effect the accuracy of MSL, however with this particular dataset, the error rates of MSL are not bad, as they range from 0.02 to 0.12.

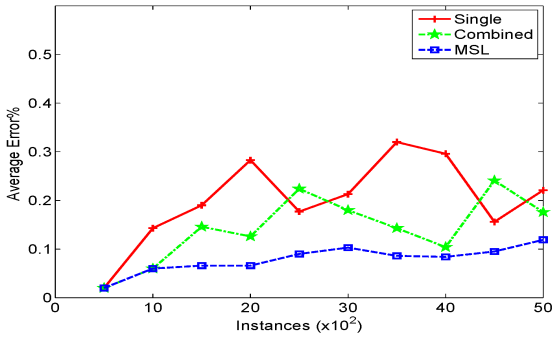

Figure 7: Experiments on Moving Hyperplane data.

5 CONCLUSIONS

In this paper, we introduced the technique MSL (Multi-Steps Learning) to facilitate effective supervised learning on evolving data with concept drift. Our main contribution is exploring a meta-learning approach to cope with concept drift, which is different to most previous techniques that use ensem-

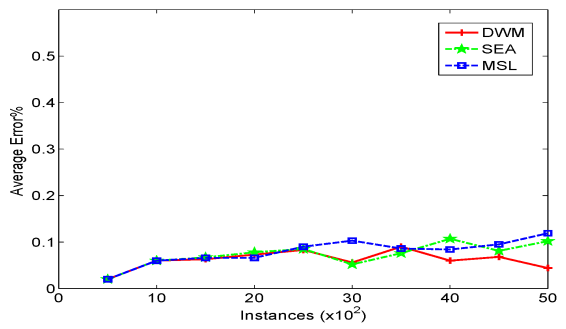

Figure 8: Experiments on Moving Hyperplane data.

ble approaches. MSL constructs its base learners in a recursive fashion, thus one MSL learner could hierarchically consist of several trained MSL learners. The number of learners generated is a function of the number of concepts hidden in the evolving data. This allows a user to trace how many concepts appear at a certain time period. MSL selects the best learner to perform the final prediction, which is also different to ensemble approaches that make majority decisions. Selecting the best one or using majority decision is a problem that has been studied by Dzeroski and Zenk (Dzeroski and Zenko, 2004).

The proposed method addresses concept drift in data, it would be interesting to investigate how this method works on the real datasets that contain concept drifts.

ACKNOWLEDGEMENTS

This work received support from the CEARTAS and SaaSiPA (Software as a Service implementation of Predictive Analytics) projects funded by Enterprise Ireland, references IP-2009-0320, CFTD-2007-0225.

REFERENCES

Bach, S. H. and Maloof, M. A. (2008). Paired learners for concept drift. In *2008 Eighth IEEE International Conference on Data Mining*, pages 23–32.

Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., and Gavalda, R. (2009). New ensemble methods for evolving data streams. In *KDD09, June 28, 2009, Paris, France*. ACM Press.

Breiman, L. (1996). Bagging Predictors. In *Machine Learning*, pages 123–140.

Dzeroski, S. and Zenko, B. (2004). Is combining classifiers better than selecting the best one? *Machine Learning*, 54:255–273.

Freund, Y. and Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explorations*, 11.

Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In *KDD'01, San Francisco, CA*. ACM Press.

Kittler, J., Hatef, M., Duin, R. P. W., and Matas, J. (1998). On Combining Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:226–239.

Kolter, J. and Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, 8:2755–2790.

Polikar, R., Udpa, L., Udpa, S. S., and Honavar, V. (2001). Learn++: An Incremental Learning Algorithm for Supervised Neural Networks. *IEEE Trans. on Systems, Man and Cybernetics. Part C*, 31:497–508.

Schlimmer, J. C. and Granger, J. (1986). Incremental learning from noisy data. *Machine Learning*, 1:317–354.

Street, W. N. and Kim, Y. (2001). A streaming ensemble algorithm (sea) for large-scale classification. In *Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 377–382. ACM Press.

Tsymbal, A. (2004). The problem of concept drift: Definitions and related work. *Computer Science Department, Trinity College Dublin, Technical Report*.

Widmer, G. (1997). Tracking context changes through meta-learning. *Machine Learning*, 27:259–286.

Widmer, G. and Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23:69–101.