The problem of concept drift: definitions and related work

Alexey Tsymbal
Department of Computer Science
Trinity College Dublin, Ireland
tsymbalo@tcd.ie

April 29, 2004

Abstract

In the real world concepts are often not stable but change with time. Typical examples of this are weather prediction rules and customers' preferences. The underlying data distribution may change as well. Often these changes make the model built on old data inconsistent with the new data, and regular updating of the model is necessary. This problem, known as *concept drift*, complicates the task of learning a model from data and requires special approaches, different from commonly used techniques, which treat arriving instances as equally important contributors to the final concept. This paper considers different types of concept drift, peculiarities of the problem, and gives a critical review of existing approaches to the problem.

1. Definitions and peculiarities of the problem

A difficult problem with learning in many real-world domains is that the concept of interest may depend on some *hidden context*, not given explicitly in the form of predictive features. A typical example is weather prediction rules that may vary radically with the season. Another example is the patterns of customers' buying preferences that may change with time, depending on the current day of the week, availability of alternatives, inflation rate, etc. Often the cause of change is hidden, not known a priori, making the learning task more complicated. Changes in the hidden context can induce more or less radical changes in the target concept, which is generally known as *concept drift* (Widmer and Kubat, 1996). An effective learner should be able to track such changes and to quickly adapt to them.

A difficult problem in handling concept drift is distinguishing between true concept drift and noise. Some algorithms may overreact to noise, erroneously interpreting it as concept drift, while others may be highly robust to noise, adjusting to the changes too slowly. An ideal learner should combine robustness to noise and sensitivity to concept drift (Widmer and Kubat, 1996).

In many domains, hidden contexts may be expected to recur. Recurring contexts may be due to cyclic phenomena, such as seasons of the year or may be associated with irregular phenomena, such as inflation rates or market mood (Harries and Sammut, 1998). In such domains, in order to adapt more quickly to concept drift, concept

descriptions may be saved so that they could be reexamined and reused later. Not many learners are able to deal with recurring contexts. Those which can include FLORA3 (Widmer and Kubat, 1993), PECS (Salganicoff, 1997), SPLICE (Harries and Sammut, 1998), Local Weights and Batch Selection (Klinkenberg, 2004).

Thus, an ideal concept drift handling system should be able to: (1) quickly adapt to concept drift; (2) be robust to noise and distinguish it from concept drift; and (3) recognize and treat recurring contexts.

2. Types of concept drift

Two kinds of concept drift that may occur in the real world are normally distinguished in the literature: (1) *sudden* (abrupt, instantaneous), and (2) *gradual* concept drift. For example, someone graduating from college might suddenly have completely different monetary concerns, whereas a slowly wearing piece of factory equipment might cause a gradual change in the quality of output parts (Stanley, 2003). Stanley (2003) divides gradual drift further into moderate and slow drifts, depending on the rate of the changes.

Hidden changes in context may not only be a cause of a change of target concept, but may also cause a change of the underlying data distribution. Even if the target concept remains the same, and it is only the data distribution that changes, this may often lead to the necessity of revising the current model, as the model's error may no longer be acceptable with the new data distribution. The necessity in the change of current model due to the change of data distribution is called *virtual concept drift* (Widmer and Kubat, 1993). Virtual concept drift and real concept drift often occur together. Virtual concept drift alone may occur, e.g. in the case of spam categorization. While our understanding of an unwanted message may remain the same over a relatively long period of time, the relative frequency of different types of spam may change drastically with time. In (Salganicoff, 1997) virtual concept drift is referred to as sampling shift, and real concept drift is referred to as concept shift. From the practical point of view it is not important, what kind of concept drift occurs, real or virtual, or both. In all cases the current model needs to be changed.

3. Systems for handling concept drift

Probably the first systems capable of handling concept drift were STAGGER (Schlimmer and Granger, 1986), FLORA (Widmer and Kubat, 1996), and IB3 (Aha et al., 1991). Three approaches to handling concept drift can be distinguished in the available systems: (1) instance selection; (2) instance weighting; and (3) ensemble learning (or learning with multiple concept descriptions).

In instance selection, the goal is to select instances relevant to the current concept. The most common concept drift handling technique is based on instance selection and consists in generalizing from a *window* that moves over recently arrived instances and uses the learnt concepts for prediction only in the immediate future. Examples of window-based algorithms include the FLORA family of algorithms (Widmer and Kubat, 1996), FRANN (Kubat and Widmer, 1994), and Time-Windowed Forgetting, TMF

(Salganicoff, 1997). Some algorithms use a window of fixed size, while others use heuristics to adjust the window size to the current extent of concept drift, e.g. "Adaptive Size" (Klinkenberg, 2004), and FLORA2 (Widmer and Kubat, 1996). Many case-base editing strategies in case-based reasoning that delete noisy, irrelevant and redundant cases are also a form of instance selection (Cunningham et al., 2003). Batch Selection of Klinkenberg (2004) may be considered as instance selection as well. Groups of instances ("batches") are considered to be relevant to the target concept if they are well classified by the current model.

Instance weighting uses the ability of some learning algorithms such as Support Vector Machines (SVMs) to process weighted instances (Klinkenberg, 2004). Instances can be weighted according to their age, and their competence with regard to the current concept. Klinkenberg (2004) shows in his experiments that instance weighting techniques handle concept drift worse than analogous instance selection techniques, which is probably due to overfitting the data.

Ensemble learning maintains a set of concept descriptions, predictions of which are combined using voting or weighted voting, or the most relevant description is selected. The first concept drift handling system STAGGER (Schlimmer and Granger, 1986) maintains a set of concept descriptions, which are originally features themselves, and more complicated concept descriptions are then produced iteratively using feature construction, the best of which are selected according to their relevance to the current data. Conceptual clustering of Harries and Sammut (1998) identifies stable hidden contexts by clustering the instances assuming that similarity of context is reflected by the degree to which instances are well classified by the same concept. A set of models is constructed then on the identified clusters. Street and Kim (2001) and Wang et al. (2001) suggest that simply dividing the data into sequential chunks of fixed size and building an ensemble on those chunks may be effective for handling concept drift. Stanley (2003) and Kolter and Maloof (2003) build ensembles of classifiers of different "age" so that each of the base classifiers sees the latest instances. All incremental ensemble approaches use some criteria to dynamically delete, reactivate, or create new ensemble members, which are normally based on the base models' consistency with the current data.

4. Base learning algorithms for handling concept drift

Many learning algorithms were used for base models in systems handling concept drift. These include rule-based learning (Schlimmer and Granger, 1986; Widmer and Kubat, 1993, 1996; Wang et al., 2003), decision trees, including their incremental versions (Harries and Sammut, 1998; Hulten et al., 2001; Street and Kim, 2001; Kolter and Maloof, 2003; Stanley, 2003; Wang et al., 2003), Naïve Bayes (Kolter and Maloof, 2003; Wang et al., 2003), SVMs (Klinkenberg, 2004), Radial Basis Functions - networks (Kubat and Widmer, 1994), and instance-based learning (Aha et al., 1991; Salganicoff, 1997; Cunningham et al., 2003).

A problem with many global eager learners (if they are not able to update their local parts incrementally when needed) is their inability to adapt to local concept drift. In the real world, concept drift may often be local, e.g. only particular types of spam

may change with time, while the others could remain the same. In the case of local concept drift, many global models are discarded simply because their accuracy on the current data falls, even if they still could be good experts in the stable parts of the data. In contrast to this, lazy learning is able to adapt well to local concept drift due to its local nature.

The advantages of lazy learning for handling concept drift were discussed in (Cunningham et al., 2003). First, lazy learning performs well with disjoint concepts, such as spam, which consists of many different sub-types; second, case-bases in lazy learning are easy to update, e.g. when new types of spam appear; and third, lazy learning allows easy sharing of knowledge for particular types of problems making easier maintaining multiple potentially distributed case-bases. Instance-based learning is sometimes criticized in that, as non-parametric learning, it needs relatively more instances to get high classification accuracy (Widmer and Kubat, 1996). However, often it is not a problem in practice, as enough instances are available.

The first instance-based technique capable of handling concept drift is IB3 (Aha et al., 1991). For each case, IB3 calculates the percentage of correct classification attempts and compares it with its class's frequency to determine which cases to keep, discarding noisy and outdated cases. IB3 was criticized as it is able to adapt to gradual concept drifts only, and its adaptation is relatively slow (Widmer and Kubat, 1996). Local Weighted Forgetting (LWF) of Salganicoff (1997) deactivates old instances, but only when similar new instances appear. Prediction Error Context Switching (PECS) is similar to LWF, but it takes into account the classification accuracy of an instance also, and is able to store instances for further reactivating (Salganicoff, 1997). PECS and LWF were shown to perform better than a simple window-based technique TWF, and PECS was the best technique overall.

5. Datasets for testing systems handling concept drift

The most popular *benchmark data* for testing concept drift handling systems is represented by the STAGGER concepts (Schlimmer and Granger, 1986) including three simple Boolean concepts of three features taking on three feature values each. It was used to test most of the systems: (Schlimmer and Granger, 1986; Widmer and Kubat, 1993, 1996; Kubat and Widmer, 1994; Harries and Sammut, 1998; Stanley, 2003; Kolter and Maloof, 2003). Another popular benchmark problem is represented by a moving hyperplane (Hulten et al., 2001; Street and Kim, 2001; Kolter and Maloof, 2003; Wang et al., 2003). The STAGGER and Hyperplane problems allow controlling the type and rate of concept drift, context recurrence, presence of noise, and irrelevant attributes. However, they do not allow checking the algorithms' scalability to large problems, which is important as concept drift mostly occurs in big amounts of data arriving in the form of stream. Some real-world problems were used to test concept drift handling systems as well: flight simulator data (Harries and Sammut, 1998), Web page access data (Hulten et al., 2001), the Text Retrieval Conference (TREC) data (Lanquillon, 1999; Klinkenberg, 2004), credit card fraud data (Wang et al., 2003), breast cancer, anonymous Web browsing, and US Census Bureau data (Street and Kim, 2001), and e-mail data (Cunningham et al., 2003). An important problem with

most of the real-world data sets is that there is little concept drift in them, or concept drift is introduced artificially, e.g. by restricting the subset of relevant topics for each particular period of time in the TREC data (Lanquillon, 1999; Klinkenberg, 2004).

6. Theoretical results in handling concept drift

The task of learning drifting concepts has also been studied in computational learning theory. Usually some restrictions are imposed on the type of admissible concept changes to make some proofs about the learnability of drifting concepts, e.g. by limiting the rate or the extent of drift. In particular, Kuh et al. (1991) determine a maximal frequency of concept changes (rate of drift) that is acceptable by any learner, which implies a lower bound for the size of a window for drifting concepts to be learnable. Hembold and Long (1994) establish bounds on the extent of drift that can be tolerated assuming possibly permanent but very slow drift, where extent is defined as the probability that two successive concepts disagree on a random instance. They also show that it is sufficient for a learner to see a fixed number of the most recent instances (a window). These results are similar to the lower bound of Kuh et al. In practice, however, it usually cannot be guaranteed that these restrictions hold true. Also the large window sizes in the theoretical bounds would be impractical to employ.

7. Incremental (online) learning versus batch learning

Most of the algorithms for handling concept drift consider incremental (online) learning environments as opposed to batch learning. While batch systems learn by examining a large collection of instances at once and forming a single model, incremental systems evolve and update a model as new instances are processed. Incremental learning is more suited for the task of handling concept drift, as in the real life data often needs to be processed in an online manner, each time after a new portion of the data arrives. This is caused by the fact that data in many current data processing systems is organized in the form of a data stream rather than a static data repository, reflecting the natural flow of data (Street and Kim, 2001; Wang et al., 2003; Hulten and Spencer, 2003).

Batch concept drift learning was considered for the sake of simplicity in (Harries et al., 1998; Klinkenberg, 2004). Klinkenberg (2004) discusses how these algorithms can be turned into incremental, in particular, for SVMs.

8. Criteria for updating the current model

Many algorithms for handling concept drift employ regular model updates as new data arrive. However, this can be too costly as the amount of arriving data may be overwhelming, and, for some applications such as spam categorization, user feedback is needed for labeling the data, which also requires time and other resources. A way to overcome this problem is to detect changes and adapt the model only if inevitable. Several criteria (so called “triggers”) were proposed in the literature. Lanquillon (1999) suggests two criteria for detecting changes without user feedback. The first

criterion is based on the average confidence in correct prediction of the model on new instances, and the second one observes the fraction of instances for which the confidence is below a given threshold. However, Lanquillon concludes that in real-world applications changes are usually much slower and less radical than those detected with the criteria and therefore more difficult to detect. Leake and Wilson (1999) suggest two similar criteria specific to case-based reasoning: (1) problem-solution regularity, and (2) problem-distribution regularity, which represent how well similarity in solutions is reflected by similarity in cases (problems), and coverage of the learning task by the case base. Although these criteria may be good measures of quality of a case-base, it is not easy to apply them in practice as triggers for model updating because the drift rate and the level of noise may vary drastically with time.

Conclusions

The problem of concept drift is of increasing importance to machine learning and data mining as more and more data is organized in the form of data streams rather than static databases, and it is rather unusual that concepts and data distribution stay stable over long period of time.

To summarize, three basic approaches to handling concept drift can be distinguished: instance selection, instance weighting, and ensemble learning. An important problem with most of the real-world datasets in existing experimental investigations is that there is little concept drift in them, or concept drift is introduced artificially. Real data including different types of concept drift are needed to experiment with proposed approaches to validate them and check their robustness to the change of different data characteristics and, in particular, their scalability. An important part of the research on concept drift is developing criteria for detecting crucial changes that allow adapting the model only if inevitable. Currently suggested “triggers” are not robust to different types of concept drift and different levels of noise, and more research is needed in this direction.

Acknowledgements. I am thankful to Prof. Pádraig Cunningham, Sarah Jane Delany and Deirdre Hogan of Department of Computer Science, Trinity College Dublin, Ireland, and Prof. Seppo Puuronen and Mykola Pechenizkiy of Department of Computer Science and Information Systems, University of Jyväskylä, Finland for discussions and comments on early versions of this paper. This material is based upon works supported by the Science Foundation Ireland under Grant No. S.F.I.-02IN.1I111.

References

1. Aha D.W., Kibler D., Albert M.K., Instance-based learning algorithms, Machine Learning, 6 (1), 1991, 37-66.
2. Cunningham P., Nowlan N., Delany S.J., Haahr M., A case-based approach to spam filtering that can track concept drift, Proc. ICCBR-2003 Workshop on Long-Lived CBR Systems, 2003.
3. Harries M., Sammut C., Horn K., Extracting hidden context, Machine Learning, 32(2), 1998, 101-126.

4. Hulten G., Spencer L., Domingos P., Mining time-changing data streams, Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD-2001, ACM Press, 2001, 97-106.
5. Kolter J.Z., Maloof M.A., Dynamic weighted majority: a new ensemble method for tracking concept drift, 3rd IEEE Int. Conf. on Data Mining ICDM-2003, IEEE CS Press, 2003, 123-130.
6. Kubat M., Widmer G., Adapting to drift in continuous domains, Tech. Report ÖFAI-TR-94-27, Austrian Research Institute for Artificial Intelligence, Vienna, 1994.
7. Lanquillon C., Renz I., Adaptive information filtering: detecting changes in text streams, Proc. 8th Int. Conf. on Information and Knowledge Management CIKM-1999, ACM Press, 1999, 538-544.
8. Leake D.B., Wilson D.C., When experience is wrong: examining CBR for changing tasks and environments, Proc. 3rd Int. Conf. on Case-Based Reasoning ICCBR-1999, Springer-Verlag, Lecture Notes in Computer Science 1650, 1999, 218-232.
9. Salganicoff M., Tolerating concept and sampling shift in lazy learning using prediction error context switching, AI Review, Special Issue on Lazy Learning, 11 (1-5), 1997, 133-155.
10. Schlimmer J.C., Granger R.H., Incremental learning from noisy data, Machine Learning, 1(3), 1986, 317-354.
11. Stanley K.O., Learning concept drift with a committee of decision trees, Tech. Report UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, USA, 2003.
12. Street W., Kim Y., A streaming ensemble algorithm (SEA) for large-scale classification, Proc. 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD-2001, ACM Press, 2001, 377-382.
13. Wang H., Fan W., Yu P.S., Han J., Mining concept-drifting data streams using ensemble classifiers, Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD-2003, ACM Press, 2003, 226-235.
14. Widmer G., Kubat M., Effective learning in dynamic environments by explicit context tracking, Proc. 6th European Conf. on Machine Learning ECML-1993, Springer-Verlag, Lecture Notes in Computer Science 667, 1993, 227-243.
15. Widmer G., Kubat M., Learning in the presence of concept drift and hidden contexts, Machine Learning, 23 (1), 1996, 69-101.
16. Klinkenberg R., Learning drifting concepts: example selection vs. example weighting, Intelligent Data Analysis, Special Issue on Incremental Learning Systems Capable of Dealing with Concept Drift, 8 (3), 2004 (to appear).
17. Helmbold D.P., Long P.M., Tracking drifting concepts by minimizing disagreements, Machine Learning, 14(1), 1994, 27-45.
18. Kuh A., Petsche T., Rivest R.L., Learning time-varying concepts, Advances in Neural Information Processing Systems (NIPS) 3, Morgan Kaufmann, 1991, 183-189.