Challenges on Classifying Data Streams with Concept Drift

Eduardo Victor Lima Barboza[1], **Paulo Ricardo Lisboa de Almeida**[1]

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)
Caixa Postal 19.081 – 81.531-980 – Curitiba – PR – Brazil

{evlbarboza,paulo}@inf.ufpr.br

Nível: Mestrado

Ingresso: 02/2022

Previsão para defesa do mestrado: 02/2024

Etapas concluídas: Revisão bibliográfica preliminar

Abstract. *Concept Drift is a common problem when we are working with Machine Learning. It refers to changes in the target concept over time, which may deteriorate the model's accuracy. A recurrent problem on concept drift is to find datasets that reflect real-world scenarios. In this work, we show some datasets known to have Concept Drift, and propose changes in an existing method (Dynse), which include making it capable of handling data streams, instead of batches, and putting some trigger on it, to make its window adaptive by detecting concept drift.*

Resumo. *Concept Drift é um problema comum quando estamos trabalhando com Aprendizado de Máquina. Refere-se a uma mudança de conceito em um intervalo de tempo, o que pode deteriorar a acurácia do modelo. Um problema recorrente em concept drift é achar datasets que reflitam cenários do mundo real. Neste trabalho, mostramos algumas bases de dados, onde sabe-se que existe Concept Drift, e propomos algumas mudanças em um método existente (Dynse), que inclui fazê-lo capaz de lidar com fluxos de dados, ao invés de lotes, e colocar algum gatilho nele, para deixar sua janela adaptativa, com detecção de concept drift.*

1. Introduction

Nowadays, we have data being generated at every moment. Much of this data is used to train Machine Learning (ML) models. However, it is not hard to imagine that some concept can change at any moment. We can cite changes in habits, weather, space, etc. ML models might have some way to adapt to changing environments, or they will lose performance. Such changes are known as Concept Drift, which happens when statistical properties of a target domain change over time [Lu et al. 2020].

For example, during the SARS-COV-2 pandemic, people began wearing masks. Think of a face detection model, which never had seen people wearing masks before. It must adapt to this new concept to maintain good performance detecting faces. In the cyber security context, [Jordaney et al. 2017] shows how these changes impact classifying malware, which is constantly updating. The same happens in detecting e-mails as spam [Kuncheva 2004], which varies according to user preferences, and spam methods are constantly changing.

We can cite more examples, like the concept drift on recommendation systems [Lu et al. 2020], which is constantly changing due to changes in habits or user preferences. Predicting energy demand is also a challenging task, as the price fluctuates by aspects that may change supply and demand, like the weather [Ditzler et al. 2015].

This paper aims to describe some datasets present in the literature on concept drift and propose changes in an existing framework for dealing with concept drift called Dynse [Almeida et al. 2018]. Such proposals include making it suitable to receive data streams of individual samples (instead of batches as in the original work) and include triggers to automatically define the accuracy estimation window size, which is used to select classifiers.

This paper is organized as follows: Section 2 describes what is a Concept Drift and its types. Section 3 presents some works related to our proposal. Section 4 shows some datasets utilized by authors that want to validate their methods. In Section 5, we explain the Dynse framework and give some proposals for improving it to be more suitable for the Data Streams scenarios. Finally, Section 6 concludes the paper.

2. Concept Drift

In the Data Stream scenario, data may arrive continuously, creating a series of challenges, which include changes in data distribution over time. Such changes are called Concept Drift and might lead to model deterioration. As stated before, concept drift happens when the statistical properties of a target domain change over time [Lu et al. 2020]. These changes might appear in different ways.

Consider $P_t(y)$ and $P_t(\mathbf{x})$ as the a priori class probabilities and the unconditional distribution at time t, respectively. A virtual concept drift occurs between timestamps t and $t + \delta$, with $\delta \geq 1$, when $P_t(y) \neq P_{t+\delta}(y)$ and/or $P_t(\mathbf{x}) \neq P_{t+\delta}(\mathbf{x})$, without changing *a posteriori* probabilities $P(y|x)$. Real concept drift happens when there is a change in the *a posteriori* probabilities, that is, $P_t(y|\mathbf{x}) \neq P_{t+\delta}(y|\mathbf{x})$, followed or not by a virtual concept drift [Almeida et al. 2020, Gama et al. 2014].

We can also differ between some types of concept drift when we look at how it happens in a timestamp. We can have sudden, gradual, incremental and reoccurring

concept drifts, as exemplified in Figure 1 [Lu et al. 2020, Ditzler et al. 2015].

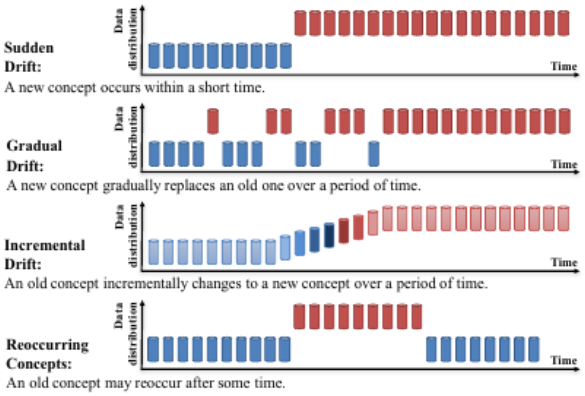

Figure 1. Types of Concept Drift. Adapted from [Lu et al. 2020].

3. Related Work

We can cite two basic approaches to deal with concept drift: Active and Passive [Ditzler et al. 2015]. Active ones try to detect concept drift and then adapt the model to a new concept, like Drift Detection Method (DDM) [Gama et al. 2004], which monitors the output error and trains a new model between a warning level and a drift level. Early Drift Detection Method (EDDM) [Baena-García et al. 2006] uses the same idea as DDM, but it takes into account the distance between concepts. Adaptive Windowing (ADWIN) [Bifet and Gavaldà 2007] tries to make an adaptive window by testing statistical differences between an older and a newer window. [Yang and Shami 2021] proposed Optimized Adaptive and Sliding Window (OASW), a drift adaptation algorithm, which combines ideas from sliding and adaptive window-based methods.

Passive methods adapt the ML model without mattering if there is a change. An example is FLORA [Kubát 1989], which maintains a fixed-size window, and whenever a new example arrives, the oldest one is forgotten. Hoeffding Trees [Domingos and Hulten 2000] are also in this group, as there is no mechanism for detecting drift.

Ensemble methods are also popular, like Oza Bagging [Oza 2005], Leveraging Bagging [Bifet et al. 2010] and the Dynse Framework [Almeida et al. 2018]. Such methods may be passive or active, and usually have a better performance on datasets with concept drift. In [Almeida et al. 2018], a passive approach that dynamically selects classifiers according to the current concept and neighborhood of the test instance is proposed. [Kozal et al. 2021] proposed an adaptive chunk size, where new classifiers are trained in new blocks and added to the group of classifiers, and obsolete models are removed.

4. Datasets

Datasets are a critical part of testing methods for dealing with concept drift. Many authors have collected or generated datasets for validating their methods. We can divide them into two: Synthetic Datasets and Real-world Datasets. A way to test if a method is suitable for dealing with concept drift is by comparing it to traditional ML models. This might also be a way to check if a dataset contains concept drift [Almeida et al. 2020].

This section presents some datasets from the literature on concept drift. These datasets will be used for validating our proposal, described in Section 5. Further research will also be done to collect more datasets.

4.1. Synthetic Datasets

Synthetic datasets are mainly used because it is possible to control the type of drift happening, making it easier to understand how the method behaves in different scenarios.

A widely used Synthetic dataset is STAGGER [Schlimmer and Granger 1986], which instance space has the attributes $size \in \{small, medium, large\}$, $color \in \{red, green, blue\}$ and $shape \in \{square, circular, triangular\}$. It has three different concepts, which follows the rules: $size = small \wedge color = red$, $color = green \vee shape = circular$ and $size = (medium \vee large)$.

The SEA concepts dataset [Street and Kim 2001] contains 60,000 random points generated with three features. Each feature has values between 0 and 10, and only the first two features are relevant (the third one is noise). This dataset is divided into four different concepts. Each data point in each block has its class 1 if $f1 + f2 \leq \theta$, where $f1$ and $f2$ are the important features. θ is a threshold defined in each concept. The authors used the values 8, 9, 7, and 9.5 for each concept.

The Rotating Hyperplane dataset was proposed by [Hulten et al. 2001]. In this dataset, a d-dimensional hyperplane is a set of points x that satisfy

$$\sum_{i=1}^{d} w_i x_i = w_0 \tag{1}$$

where x_i is the ith coordinate of x. When $\sum_{i=1}^{d} w_i x_i \geq w_0$, instances are labeled positive, and when $\sum_{i=1}^{d} w_i x_i \leq w_0$, they are labeled negative. Time-changing concept is simulated by changing the orientation and position of the hyperplane by changing the relative size of weights. Gradual and Incremental drift might be present in this dataset.

4.2. Real-world Datasets

While synthetic datasets can be used to develop the approaches and check their performances under controlled environments, real-world datasets are imperative to check the approach's quality under real operation circumstances.

[Müller and Salathé 2020] collected data from Twitter posts about opinions of people on vaccination. It contains 11,893 annotations, divided into 13 bins of 90 days each. The authors argue that a crisis, like a pandemic, might greatly influence people's opinion, which can be put as a Concept Drift.

It is also possible to observe concept drift on image data. [Almeida et al. 2018] proposed the use of a parking lot images dataset, which the authors called PKLot. There are images in different concepts of weather, camera angles, and lighting. It contains 105,845, which of those 43.48% were labeled as occupied and 56.42% empty.

[Souza et al. 2020] presented the Insects dataset, where the authors created a trap for capturing insects using a sensor. Authors say that many aspects might influence the behavior of insects. Temperature, for example, influences their metabolism. For the sensor to detect suitable insects, it needs a model that can adapt to new concepts. There

are 905,145 instances, which the authors have divided them into different datasets with different types of concept drift.

5. The Dynse Framework

Dynse was proposed by [Almeida et al. 2018]. It is a method for dealing with Concept Drift based on the dynamic selection of classifiers. The authors mentioned that many methods might be used for selecting classifiers, but they proposed the KNORA-ELIMINATE (KNORA-E) [Woods et al. 1997] method as a default configuration. In general, it works by selecting the classifiers that better perform classifying data instances in the neighborhood of the test instance.

As new batches of labeled instances become available, new classifiers are added to a pool P. The pool should be as big as possible, as with more classifiers trained at different moments, we might have better ensembles to classify new instances. However, this pool might need to be pruned due to constraints like time and memory. One way to do this is by taking off the worst-performing classifier, or the oldest one.

A window containing the N latest labeled batches, where N is a parameter of the method, is used to select classifiers for the current test instance dynamically. So this is a batch-learning algorithm, as it periodically needs a batch of new instances. Moreover, it is also a passive approach to dealing with Concept Drift, as there is no mechanism for actively detecting drift (the window is constantly updated as new labeled batches become available). Figure 2 shows a scheme of how Dynse works.

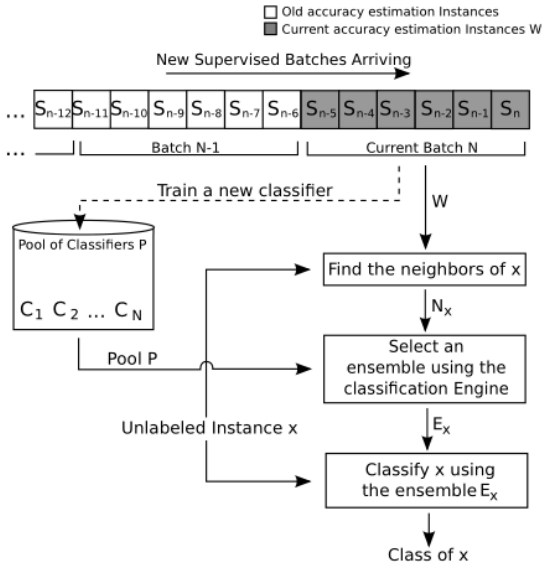

Figure 2. Dynse Scheme. Adapted from [Almeida et al. 2016]

5.1. Modification Proposal

Our current work is focused on two modifications for the Dynse Framework. First, the Dynse uses a labeled stream of batches over time to train new classifiers, but it cannot receive data streams of labeled (single) instances. So, the first proposal is to make it able to deal with Data Streams.

The second one lies in the accuracy estimation window it needs to estimate the classifier's competence (current batch N in Figure 2). The user must define its size, which may cause some issues. For example, the window can be too short and not have enough data for generalization. If we can make this window adaptive, we might get better adaptation to concept drifts. For this, we can add some triggers, like DDM or EDDM. This way, the window size could increase under stable periods and quickly shrink when a concept drift is detected.

Differently from methods that rely purely on triggers to adapt to concept drifts, the Dynse framework may be more robust to false alarms, since the trigger will be used to shrink the accuracy estimation window without discarding the classifiers in the pool. By making the Dynse able to process new instances whenever they arrive, and making its window adaptive, we hope to get faster adaptations when concept drift occurs, combined with better accuracies under stable periods. To validate these proposals, we we will compare statistical metrics (e.g. accuracy, f1-score) between Dynse and other methods in the literature.

6. Conclusion

Concept Drift is a common problem for streaming data, where the data pattern might change over time. There is a lack of datasets to test the ability of the methods to adapt concept drifts. The literature shows only a handful of datasets (synthetic and real-world ones); thus, a deeper search for datasets, or the proposal of new ones, is one challenge to overcome in this work.

We proposed possible changes in the Dynse framework, including the ability to process data streams and the use of triggers to adapt its accuracy estimation window. As discussed in this work, some triggers that may be used to adapt this window include the DDM and the EDDM methods. These improvements may increase the Dynse performance under both stable and drifting periods. This is a work in progress, and in the end we hope to have an improved version of Dynse, faster to adapt do concept drifts, and more suitable to deal with streams of data.

References

Almeida, P. R., Oliveira, L. S., Britto, A. S., and Sabourin, R. (2018). Adapting dynamic classifier selection for concept drift. *Expert Systems with Applications*, 104:67–85.

Almeida, P. R. L. d., Oliveira, L. S., Britto, A. D. S., and Sabourin, R. (2016). Handling concept drifts using dynamic selection of classifiers. In *28th ICTAI*, pages 989–995.

Almeida, P. R. L. d., Oliveira, L. S., Souza Britto, A. d., and Paul Barddal, J. (2020). Naïve approaches to deal with concept drifts. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1052–1059.

Baena-García, M., Campo-Ávila, J., Fidalgo-Merino, R., Bifet, A., Gavald, R., and Morales-Bueno, R. (2006). Early drift detection method. *4th WKDDS*.

Bifet, A. and Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In *SDM*.

Bifet, A., Holmes, G., and Pfahringer, B. (2010). Leveraging bagging for evolving data streams. In Balcázar, J. L., Bonchi, F., Gionis, A., and Sebag, M., editors, *Machine*

Learning and Knowledge Discovery in Databases, pages 135–150, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ditzler, G., Roveri, M., Alippi, C., and Polikar, R. (2015). Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25.

Domingos, P. and Hulten, G. (2000). Mining high-speed data streams. *Association for Computing Machinery*, page 71–80.

Gama, J., Medas, P., Castillo, G., and Rodrigues, P. (2004). Learning with drift detection. In Bazzan, A. L. C. and Labidi, S., editors, *Advances in Artificial Intelligence – SBIA 2004*, pages 286–295, Berlin, Heidelberg. Springer Berlin Heidelberg.

Gama, J. a., Žliobaitundefined, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4).

Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In *7th SIGKDD*, KDD '01, page 97–106, New York, NY, USA. ACM.

Jordaney, R., Sharad, K., Dash, S. K., Wang, Z., Papini, D., Nouretdinov, I., and Cavallaro, L. (2017). Transcend: Detecting concept drift in malware classification models. In *26th USENIX*, pages 625–642, Vancouver, BC. USENIX Association.

Kozal, J., Guzy, F., and Woźniak, M. (2021). Employing chunk size adaptation to overcome concept drift.

Kubát, M. (1989). Floating approximation in time-varying knowledge bases. *Pattern Recognition Letters*, 10(4):223–227.

Kuncheva, L. I. (2004). Classifier ensembles for changing environments. In Roli, F., Kittler, J., and Windeatt, T., editors, *Multiple Classifier Systems*, pages 1–15, Berlin, Heidelberg. Springer Berlin Heidelberg.

Lu, J., Liu, A., Song, Y., and Zhang, G. (2020). Data-driven decision support under concept drift in streamed big data. *Complex & Intelligent Systems 6*, pages 157–163.

Müller, M. and Salathé, M. (2020). Addressing machine learning concept drift reveals declining vaccine sentiment during the covid-19 pandemic.

Oza, N. (2005). Online bagging and boosting. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2340–2345 Vol. 3.

Schlimmer, J. C. and Granger, R. H. (1986). Incremental learning from noisy data. *Mach. Learn.*, 1(3):317–354.

Souza, V. M. A., dos Reis, D. M., Maletzke, A. G., and Batista, G. E. A. P. A. (2020). Challenges in benchmarking stream learning algorithms with real-world data. *Data Mining and Knowledge Discovery*, 34(6):1805–1858.

Street, W. N. and Kim, Y. (2001). A streaming ensemble algorithm (SEA) for large-scale classification. In *7th SIGKDD*, KDD '01, page 377–382, New York, NY, USA. ACM.

Woods, K., Kegelmeyer, W., and Bowyer, K. (1997). Combination of multiple classifiers using local accuracy estimates. *Pattern Analysis and Machine Intellig.*, 19(4):405–410.

Yang, L. and Shami, A. (2021). A lightweight concept drift detection and adaptation framework for IoT data streams. *IEEE Internet of Things Magazine*, 4(2):96–101.