Concept Drifts - Notes

Matheus Rosso

Contents

1 Discussion and Review on Evolving Data Streams and Concept Drift Adapting - Khamassi et al (2016)

- Link to access the paper.
- **Concept drift** occurs when a data stream evolves over time, so its stochastic generation process is non-stationary.
- The paper defines the problem of concept drift and presents a taxonomy for understanding most relevant (until then) approaches for handling it.
- **Data streams** imply in three major difficulties for computing and processing: i) unboundedness of size; ii) high steady rate of arriving; iii) non-stationarity of data.
- Data streams require online processing of data, which differs from offline learning in which all data is stored in memory during model training. **Online learning**, in its turn, proceeds to the parameters estimation at each received observation (or set of observations). Data stream also affect how predictions should be produced: instead of on demand, predictions are required to be a fast response to passed data.
- Not only data streams impose a new paradigm for learning, but also it highlights the problem of non-stationarity - changes in the distribution of variables involved in model estimation and prediction.
- Considering online machine learning, its standard algorithms are not able to adequately handle concept drift, since they do not accomplish unlearning of outdated concepts. Differently, **adaptive and evolving learning algorithms** can discard old data and focus on current concepts that better depict the underlying population.
- **Adaptive learning algorithms** consider the discard of data for model training, so this self-configuration allows the learning process to react against concept changes. Three main issues should be addressed by adaptive learning: i) the monitoring of concept drifts; ii) the selection of which data should be kept or discarded; and iii) the mechanism under which parameters and structure change in response to concept drift.
- **Evolving learning models** go beyond incremental algorithms since they capture concept drifts and intervene over their structures so a new model is constructed based on the past one and considering

the new environment.

- **Concept drift:** given a classification task where the response variable is $Y \in \{y_1, y_2, ..., y_C\}$ and the vector of inputs is X, the distribution of main interest is the *joint distribution* $P(X, y_c)$. For a give time period t, $D = \{P(X, y_1), ..., P(X, y_C)\}$ is named a *concept.* There is concept drift when for two distinct time periods t and t': $P_t(X, y_c) \neq P_{t'}(X, y_c)$ for a given class c.
  - Learning models usually are trained to classify by maximizing the posterior probability $P(y_c|X)$.
  - Since $P(X, y_c) = P(X|y_c).P(y_c)$, the Bayes theorem implies:

    $$P(y_c|X) = \frac{P(X|y_c).P(y_c)}{P(X)} = \frac{P(X|y_c).P(y_c)}{\sum_{c=1}^{C} P(X|y_c)P(y_c)} \tag{1}$$

    Given that the learning task at hand is supervised and $P(X)$ is constant over Y, a concept drift may occur as a consequence of a change in $P(y_c|X)$, or in $P(X|y_c)$, or in $P(y_c)$.
  - **Real concept drift:** changes in $P(y_c|X)$ over time are maybe the most relevant source of concept drift, since this directly impacts the decision boundary and, as a result, deteriorates the performance of models. Therefore, performance metrics are the best way to track real concept drifts.
  - **Virtual concept drift:** this type of concept drift refers to changes in the class-conditional probability $P(X|y_c)$ without effects on $P(y_c|X)$, so the decision boundary remains the same. To handle such concept drifts, class-conditional distribution of inputs X should be be monitored.
  - **Class prior concept drift:** refers to changes in $P(y_c)$, which may not affect the decision boundary when there is only class imbalance or may affect the decision boundary if a new class emerges or when a previously existing one no longer exists.
  - **Concept drift characteristics:**
    - **Speed of drift**: the duration of drift is the amount of time between a given concept and a new one. Speed is considered as the inverse of drifting time, and can be either abrupt or gradual. An **abrupt drift** is more easily identified, since it suddenly deteriorates performance. A **gradual drift** is harder to detect, since takes more time to evolve from one concept to another.
    - **Drift nature:** a *gradual drift*, in its turn, can be either probabilistic or continuous. **Gradual probabilistic drift** refers to a case when, during the transition phase, samples

    can be drawn according to two distinct concepts both with positive probabilities. **Gradual continuous drift** implies in slight changes during the transition phase.

    * **Severity of drift:** according to the amount of changes produced by a drift, it can be local or global. A **local concept drift** applies to only subsets of the inputs space, so the detection is even harder: changes in concept may be taken as noise, and vice-versa, so the adaptive learning model should precisely separate drifts from noises and deal of the scarcity of instances that depict a drift. A **global concept drift** applies to the entire inputs space, so it is more evident to notice.
    * **Drift recurrency:** refers to how much recurrent a concept can be over time. Some concepts may reappear periodically (**cyclical recurrent drift**) or without periodicity (**acyclical recurrent drift**), or even it may be no recurrent at all.
    * **Drift predictability:** a concept drift may occur with some underlying pattern or completely by chance. **Predictable drifts** are faster and easier to be detected and handled.

- The design of the most appropriate approach for concept drift handling depends on main characteristics of the machine learning system at hand. In addition to that, the definition of an approach should raise the following questions regarding the ways under which concept drifts should be detected and handled.

  - How are data processed in the application?
  - How is learning task processed?
  - How is concept drift monitored?
  - How is concept drift handled?
  - What are the performance criteria to be considered?

- **Data process:** there are two main alternatives under which data can be processed within a machine learning system: *sequentially* (single instance at a time) or through *window of data* (multiple instances at a time).

  - **Sequential methods:** these are suitable for sequential processing of data, and are inspired by online statistical analysis for anomaly/change detection.

* Considering its definitions and syntax, given a set of random observations $\{X_1, X_2, ..., X_N\}$, each may be generated by some underlying distribution D_i. Even so, if $D_i = D_0 \ \forall \ i \in \{1, 2, ..., N\}$, then the distribution is said to be **stationary**.

* However, if for a sub-sequence $\{X_1, ..., X_k\}$, with $k \in (1, N)$, there exists a change point $\lambda \in (k, N)$ such that D_0 applies for $\{X_1, ..., X_k\}$ and D_1 for $\{X_\lambda, ..., X_N\}$, with $D_0 \neq D_1$, then the distribution is **non-stationary**.

* Sequential methods for concept drift handling seek to find if there exists such change point λ for which two distinct distribution D_0 and D_1 emerge. So, under the standard **theory of hypothesis testing**, a null hypothesis $H_0 : D_0 = D_1$ stands for no concept drift, while the alternative hypothesis $H_1 : D_0 \neq D_1$ means that in some change point λ a concept drift has occurred.

* Consequently, two main statistical metrics are the **false alarm rate** $\alpha = P(H_1|H_0)$, and the **missed detection rate** $\beta = P(H_0|H_1)$.

* In order to assess if the null hypothesis should be rejected, a **dissimilarity measure** $D_\lambda(D_0, D_1)$ should be calculated, so the change can be said to be statistically significant or not. $D_\lambda(D_0, D_1)$ can be calculated directly onto the data by measuring the distance between incoming instances and a set of past observations. Alternatively, $D_\lambda(D_0, D_1)$ may follow from the comparison between statistics from two distinct collections of data.

* In addition to the dissimilarity measure, a **change threshold** τ is used to identify significant values of $D_\lambda(D_0, D_1)$. This threshold can be either **fixed** or **variable**. A large fixed value is suited for *capturing gradual drifts*, while small fixed values are more appropriate for the *detection of abrupt drifts*. Variable thresholds may be better for handling drifts when their types can not be foreseen, but α and β rates should be estimated and controlled to check for the health of the drift monitoring.

* **Criticisms:** sequential approaches are memoryless, since they discard observations once they are checked against a reference data to assess for the occurrence of concept drift. As a result, *they are more suited for detecting abrupt then gradual drifts*. Another point concerns the relevance of accomplishing the *temporal dependence* that may apply for data streams - windowing methods are inherently ready for considering time series components during concept drift handling.

- **Windowing methods:** this approach for drift handling is not memoryless, though has short memory. So, it considers the most recent observations as those that are the most informative to suggest which is the current data distribution, and if there has been a concept drift. **Windows of data** are representative collections of data, raw or processed (statistics from data or model performance), that refer to distinct periods of time. They can be understood based on four dimensions: specificity, nature, size, and positioning strategy.
  - **Specificity:** can be either **inside the learner**, where the windowing method refers to and is integrated into an exclusive statistical learning method (monitoring metrics specific for it), or **outside the learner**, where the method is generic, focusing on general performance metrics or data distribution statistics.
  - **Nature: data-based windows** are defined by the number of instances belonging to them. **Time-based windows** have definition rules based on time constraints.
  - **Size:** a **fixed size** window of small size is more suited for *capturing abrupt drifts*, while large sizes are more appropriate for *gradual drifts*. When the type of drift can not be foreseen, **variable size** windows may be a better choice.
  - **Positioning strategy:** concerns how the main window evolves over time during the drift tracking.
    - **Single window:** may be either a **sliding window**, in which the monitoring data is kept with a constant size by dropping the oldest instance at each instance arrived, or a **landmark window**, under which the set of instances increases in size until some condition is met, such as the detection of a drift.

      Note: even though the paper does not mention this issue, it is likely that the drift detection depends on the comparison of, say, descriptive statistics for the single window at time t_1 against those for the window at t_0.

      Note: the majority of considerations concerning data processing applies for drift detection mechanisms. However, some principles and structures may also apply for an online learning operation or a system of automated model training.
    - **Two windows**: here, one window serves as baseline for the current data batch.

      Separated windows: the reference is kept constant, while the current data varies throughout the time. This is particularly useful for offline learning, since current data can be compared against the training data. With respect to concept drift types, separated windows are suited for *detecting gradual drifts*.

Adjacent windows: baseline and current data batches are consecutive, which is designed for concept drift detection of online learning systems. Sizes of both windows may be either: **fixed/fixed**, useful for *abrupt drift detection*; **variable/fixed**, in which baseline data increases until a drift is found (useful for *detecting gradual drift*); **variable/variable**, more flexible approach under which baseline and current data are constrained so that dissimilarity measure can be maximized.

Overlapping windows: both windows have data in common. Useful when data is scarce.

Discontinued windows: only subsets of reference data batch is used to comparison with current data. Adequate for *detecting local drifts*.

      · There is also the possibility of exploring more than two windows at once.

    * **Criticism:** in general, windowing methods treat data uniformly, so it is more adequate to capture global concept drifts.

  - **Note:** the way how data is processed in a machine learning system creates constraints to how a concept drift handling approach can process data to detect relevant changes in data distributions. Even so, given such restrictions, there are several alternatives for the data monitoring task as presented above.

- **Learning process:** the configuration of a machine learning system will guide how concept drifts can be identified and treated. This is particularly true regarding if the learning process relies on a *single learner* or an *ensemble of learners*. Moreover, when designing a machine learning system, one may consider some characteristics of these two alternatives that help the most adequate choice when it comes to a proper concept drift handling.

  - **Single learner:** single incremental algorithms can be improved to handle concept drift by adding **forgetting mechanisms** that discard outdated instances from the training data. Another possible improvement to incremental learning algorithms is to convert them to be **self-adaptive**, which refers to automatically change their structure as soon as some drift is detected. This leads to two questions: how to define and track concept drifts and how to incorporate such changes into the algorithms. Irrespective of how single learners are adapted to conceal concept drifts, they may not respond well to recurrent drifts as they do not make use of information concerning already seen concepts.

- **Ensemble learners:** composing a committee of base learners is expected to increase the generalization capacity of a machine learning application, as for an individual prediction one poor base learner can be mitigated by another base learner. This is specially helpful for reducing the variance component of the expected prediction error. Even more, ensembles of learners are suited for handling concept drifts, which derives from two properties that they are likely to have: *diversity* and *adaptability*. The following three dimensions of a machine learning system can be used for controlling the extent to which those attributes apply.
    - **Training data management:** how available data is shared among base learners may help to assure good levels of diversity and adaptability for the ensemble.
        - **Block-based technique:** this approach defines the training data by its length, so whenever the data available for training reaches a given volume, this batch may be used for re-training or replacing the base learner with the poorest current performance. This method is appropriate for *handling gradual continuous drifts*, since the ensemble is made up of base learners trained on data from different period of time. This method is also useful when true labels arrive with a moderate-to-high lag. The challenge of using block-based technique is to regulate its size in order to balance between accuracy (large batches improve performance) and drift reaction (small batches react to changes more rapidly).
        - **Weighting data technique:** all base learners may be trained using the same set of instances, but all of them receives different weights across the ensemble members. One interesting approach is, for a given learner, to define higher weights for misclassified instances or for those with higher prediction errors. This approach is suited for *handling local drifts* and for classification contexts where there is imbalanced classes. As a drawback, this strategy increases computational complexity.
        - **Filtering data technique:** subsets of the entire set of available features are randomly selected for each base learner, even that some predefined rules may help with this allocation. Preferably, each base learner can also receive only subsets of all available instances, considering similar subsets regarding observations. Additionally, an alter-

native is to split data only in terms of observations, keeping the same set of features for all ensemble members. This approach is appropriate for *handling local drifts*, and may help detecting novel classes. Drift detection is more direct here, since specific drifted features can be pointed out.

- **Structure management:** concerns the ensemble size, i.e., the number of base learners composing it.
  - **Fixed ensemble size:** this fixed number of learners can be defined using validation techniques. Periodically, the performance of each individual learner should be assess so the weakest one can be replace by another learner, or it can be re-trained using new data. This strategy has a relatively low level of complexity.
  - **Variable ensemble size:** the structure is automatically defined, and one possible implementation of this strategy considers a pool of ensemble from which only those with best performance can effectively make part of the decision-making ensemble. Since weak learners are not dropped off, such method is beneficial for *handling recurrent drifts.* Definition of weights for all learners in the pool may follow from the optimization of the overall system performance.
- **Final decision management:** refers to how individual predictions from ensemble members are processed in order to generate a final prediction.
  - **Dynamic weighting technique:** following some predefined regularity and considering the weighting of individual predictions so they can configure a pool of votes, the accuracy of each individual learner trained on some batch of data is assessed over more recent data. Higher weights are then specified for learners with higher accuracy. This drift handling strategy is still subject to new concepts emerging subsequently to the validation data. Even so, it is appropriate for *handling gradual continuous drifts.*
  - **Dynamic selection technique:** only one base learner is effectively applied to the decision process. It can be the one trained on the most recent batch of data, or the learner with the highest evaluated accuracy, or even that one whose training data is the most similar to the incoming instance. This approach is suited for *handling either recurrent or local drifts*, besides of controlling the system complexity.
  - **Combining dynamic weighting and selecting techniques:** this strategy selects

a subset of learners, and then applies dynamic weighting for their predictions. This brings the possibility of *handling both global and local drifts, besides of the recurrent ones.*

      * Some promising aspect to be explored when developing a machine learning system that handles concept drifts is the **heterogeneity** of ensemble members, either by using different statistical learning methods or by applying different settings of hyper-parameters. Another point to have in mind is to guarantee that base learners with efficient drift handling are not dropped from the ensemble when their performances are not great. Similarly, not only expected performance, but also which types of drifts a learner can detect and handle should be considered when selecting learner types for the ensemble.

- **Monitoring process:** again, the machine learning application creates conditions on how changes can be monitored, while alternatives exist to implement drift tracking. The main aspect to have in mind is the availability of *prediction feedbacks*, which depends ultimately on the availability of true labels.

  - **Supervised monitoring:** feasible when true labels are obtained fast, and is focused on the monitoring of learners performance. Thus, is designed for *detecting real concept drifts.* In terms of classification problems, **accuracy, recall (sensitivity** and **precision** are main metrics to keep under control. Together with recall, **specificity** is useful for *tracking (class) prior concept drifts.*

  - **Unsupervised monitoring:** when prediction feedback is delayed, concept drifts can be promptly detected considering the distribution of input variables. Thus, this method is suited for *tracking virtual concept drifts.* Some methods focus on changes over time (with fixed distributions among classes), while others focus on changes in the distribution over the features space. Another collection of methods monitor the inner structure of learning methods.

  - **Semi-supervised monitoring:** combines the use of information on inputs and outputs, being appropriate for contexts when true labels are scarcely available.

- **Adapting process:** once the data and learning processes have been designed or extended to handle concept drifts, and given a choice of monitoring process, the next logical step is to define how learners will be adapted in response to changes.

- **Informed methods:** these approaches explicitly detect and inform changes in tracked relevant concepts.
  - They are crucial for contexts when concept drifts should not only imply in updates by the machine learning system, but instead the information of the occurrence of changes is also relevant for the application management and improvement.
  - Some metrics that can constitute an informed method are those concerning performance or the structure of the learners.
  - Informed methods are reactive, as they first detect a drift, then trigger alerts or communications, select data to be kept or dropped from the training set, and eventually review the learners type or structure.
- **Blind methods:** these alternatives do not communicate possible changes in the data stream, but implicitly adapt the learners to incorporate current concepts and discard the outdated ones.
  - By periodically re-training or reviewing parameters of the machine learning system, such approaches are adequate for *handling gradual continuous drifts.*
  - A possible implementation considers fixed size sliding windows of training data in a first-in-first-out manner.
  - Another implementation dynamically review the weighting of training instances, given more influence to more recent or more representative data points.
  - Dynamic learners selection, by replacing from the ensemble those learners with the poorest performance, is another possibility for continuously being up-to-date with the current concept.
- Note that blind methods are more complex both in terms of technology and computation. Additionally, they review machine learning systems even when this would not be necessary. Informed methods, in their turn, may imply in waste of resources as false alerts are produced, so false positives and also false negatives are relevant parameters when implementing them.

• **Performance criteria:** a final task when designing an approach for concept drifts handling within a machine learning application is the definition and monitoring of how the system is performing when detecting and responding to changes. Main criteria are autonomy, reliability, parameters setting and complexity. Most of them should be optimized differently according with constraints given by the application.

  - **Autonomy:** the less human interventions are expected to handle drifts the more autonomous the system will be. Data process approaches of variable change threshold (sequential method) or self-adapting windows, besides of self-configuring and self-adapting learners can conceive a high degree of autonomy for a drift handling approach.
  - **Reliability:** the more information on detected changes (their nature, intensity, time of occurrence) the more reliable the system will be. Some main aspects concerns false alarm rate, missed detection rate, and delay of detection. A drift handling approach will also be more reliable when these metrics are under control and assuming acceptable levels.
  - **Parameters setting:** this criterion relates with autonomy and complexity issues. In general, the less parameters needed to be adjusted the better.
  - **Complexity:** reflects directly how data is received and processed by the system. Thus, a drift handling approach should be as few complex as possible, given constraints imposed by volume of data and required response time.

- **Final remarks:** the paper and these notes provide taxonomy and reference points for when considering how to handle concept drifts. In addition to the definition of concept drift, main aspects of existing approaches are defined both as a function of application constraints and as alternatives to be considered, together with considerations on how these options respond to different drift types.

2 A Review on Concept Drift - Kadwe, Suryawanshi

- Link to access the paper.
- Concept drift is an issue that follows from machine learning applications where data are gathered as streams rather than static datasets. Consequently, concepts, i.e., data generating processes governing how data behave in a populational level may evolve over time (non-stationarity). These changes in distributions require that machine learning systems react to them, seeking to maintain adequate performances.

- Given the joint distribution $P_t(X, y)$ at time period t, there is concept drift when $P_t(X, y) \neq P_{t'}$ for $t \neq t'$. This may follow from changes in $P_t(y)$, $P_t(X|y)$ and $P_t(y|X)$. Concept drifts are also classified according to the speed of changes: they can be abrupt, incremental, gradual or recurrent. Approaches for handling concept drifts should take anomalies/outliers as actual concept drifts, since noise changes in distributions are expected to happen on a regular basis.

- Monitoring concept drifts may focus on data distributions (inputs, output, etc.), (estimated) relationships between features and the output, performance of models.

- Sections II and III (also part of section IV) of the paper present several different approaches to detect concept drifts, by using methods for tracking changes and by applying statistical tests in order to verify their statistical significance.

- The main purpose of distinguishing concept drifts from noises is to appropriately modify the ensemble of learners in order to guarantee that predictions are being made upon the current relationship between inputs and the output variable. Some strategies for concept drift handling are as follows:

  - Forgetting mechanisms manipulate over the weighting of training instances or individual predictors.
  - New learners are added into the ensemble as new batches of training data are available.
  - Ensemble members are retrained using new instances (online or according to batches of data).
  - Learners with worst performances are excluded from the ensemble, being replaces by new members trained on more recent data.
  - The set of input variables changes over time.