Detecting and Adapting to Concept Drift in Continually Evolving Stochastic Processes

Sunanda Gamage
rcsunanda@gmail.com

Upeka Premaratne
University of Moratuwa, Sri Lanka
upeka@ent.mrt.ac.lk

Abstract

[to fill]

1. Introduction

The vast majority of machine learning research focuses on building models (eg: regression models, classifiers) that are static; i.e. the model is trained one time on a training dataset and then applied to test data. This works fine when the underlying stochastic process that is modeled remains stationary. However, in many real world applications, the process is non-stationary, and it continuously changes with time.

For example, consider the video feed from a traffic camera, which can undergo changes from morning to night, and from summer to winter. The video feed process in this example undergoes gradual continuous change. The drift can also be abrupt and large, such as an impactful market event that affects stock prices.

The underlying stochastic process in these examples is non-stationary, which means that the joint probability distribution of the process attributes, $P(\boldsymbol{X})$, is time varying. This is known as concept drift. Applying a model that is trained one time on some initial training dataset to such a process results in performance degradation. Therefore, methods to characterize the concept drift, and adapt machine learning models to drift by incremental learning are an important research direction in machine learning.

[Summarize current the methods and their deficiencies]

In this work, we improve on the drift detection techniques used in [A] and [B], and present an efficient ensemble technique of incremental learning to model a continuously changing stochastic process. The improved drift detection technique tracks a difference metric (what metric?) between probability distributions estimated from two sample windows before and after a time point. The estimated distribution is typically the joint probability distribution of attributes $P(\boldsymbol{X})$ for the unsurprised case, or joint attribute and class distribution $P(\boldsymbol{X}, Y)$ for the supervised case. When the continuous sum of this metric since the last adaptation rises above a threshold m_{drift}, it is decided that an adaptation should be made.

At this point, a new model is learned from a suitable number of most recent samples, and it is added to an ensemble of previously learned models. The final time varying model is a weighted some of the sub models in the ensemble. The weights can be set in a way such that older sub models have less relevance (a form of forgetting). They can also be set to reflect the similarity between the probability distributions of the current sample window, and the windows corresponding to each sub model. In the latter case, the distributions relevant to each sub model also need to be stored.

Our drift detection and adaption techniques can act as a wrapper method independent of the machine learning model being employed. The drift detection technique can be made to detect different types of drift [B] by choosing the relevant distribution for computing the difference metric. The methods work well for both supervised learning models such as classifiers, or unsupervised models such as hidden Markov models. The adaptation technique does not cause catastrophic forgetting. The forgetting factor can in fact be controlled by setting appropriate weights on the sub models in the ensemble. experiment results of evaluating the methods on artificial and real-life datasets show good performance in detecting drift and adapting to it.

The rest of this paper is organized as follows. Section 2 presents related work in detecting concept drift and adaptation methods. In Section 3, the drift problem is formally defined, and solutions for drift detection and adaptation are described. In Section 4, we presents the result of evaluating the method on several artificial and real-life datasets. Conclusions and future directions are given in Section 5.

2. Related Work

2.1 Detection techniques

Several works in the literature have developed drift detection techniques. In [A], drift is measured by tracing the error rate at test time, where a rise in error is considered as a sign of drift. Two threshold levels, warning level and drift level, are predefined, and when error increases above the drift level, the detector triggers a drift action. This is method is computationally efficient, but needs the true labels at test time to compute the error rate.

A novel method introduced in [B], called concept drift maps, computes a difference metric in real-time between probability distributions (Total Variation Distance) estimated from two sample windows before and after a time point. This method allows the use of any probability distribution that represents the dataset, including the distribution of any subset of attributes. Using a distance metric instead of test error rate means it is well suited for cases where true labels are not revealed at test time.

2.2 Adaptation techniques

There is a large body of work that explores adaptation methods for concept drift. An excellent overview of these methods and their key design principles is given in [C] and [N]. A common approach is to divide the historical data into blocks or windows that represent a context (a period where no drift has occurred) and use it to train a learner [D][E]. Alternatively, training sets can be formulated by instance weighing, where weights are given according to age of instances or their performance with regard to the current concept [G][H].

The adaptation method described in [A] retrains a new model on the most recent context when a drift is detected, and any previous model is discarded. This leads to catastrophic forgetting, which means that any previously learned knowledge is completely forgotten, and recurring patterns will have no benefit from it.

In order to retain previous knowledge, [E][F] use an ensemble of models. In [F], Sequentially arriving samples are batched together by a fixed batch size, and weights of previous sub models in the ensemble are updated based on their performance on the batch. Then, the batch is sampled to extract the instances that were not classified accurately by previous sub models, and a new sub model is trained on them. This method does not detect drifts, but keeps updating the ensemble weights and adding new sub models as new batches arrive. This is computationally not efficient. Furthermore, this technique requires that true labels of new examples must be available.

Fields that are related to learning under concept drift are "domain adaptation" and "transfer learning". In these areas, the topic of adapting a model trained on one domain or task to a different one is studied. The majority of this work considers one source domain and one or a few target domains. However, some researchers have extended the adaptation to the incremental setting [J][K][L]. In [J], a method to adapt a base classifier to a change in class probability is described. The authors also present an extension for the case of sequentially arriving examples. However, this method does not work for changes in distributions of attributes or class conditional attributes.

[Our unique points, in comparison to above – elaborate on these]

- Especially good for generative models (unsupervised), since we have change detection of P(X)
    - Weight updates is made possible by using distance metric rather than classification error
- Unlike [F], we have a drift detector and it can detect multiple types of drift
- Because ours is an ensemble method, and due to distance based weighting of sub models,
    - Previously learned knowledge is not discarded
    - Different sub models can master different parts of the data distribution
    - works well for recurring concepts
- First known use of concept drift maps
- Yields well to a practical implementation and optimizations (

3. Methodology

3.1 Problem definition

Consider a stochastic process characterized by a joint distribution $P(\boldsymbol{X})$ over the random variables $\boldsymbol{X} = \{X_1, \dots, X_n\}$. If our task is classification or regression, a target variable Y is also present, and the process is represented by the joint distribution $P(\boldsymbol{X}, Y)$ over $\{\boldsymbol{X}, Y\}$. A sequence of samples generated by this process are observed.

We say a concept drift has occurred in the process between two time points t, u when $P_t(\boldsymbol{X}, Y) \neq P_u(\boldsymbol{X}, Y)$. Since it is often not practical to estimate the distribution in effect at a specific point in time, we consider drift occurring between time intervals $[t_1, t_2]$ and $[u_1, u_2]$, which is given by $P_{[t_1,t_2]}(\boldsymbol{X}, Y) \neq P_{[u_1,u_2]}(\boldsymbol{X}, Y)$.

The drift detection problem is to quantify the above change in probability distributions with time, and identify when significant drift has occurred. The adaptation problem is to make necessary changes to a learned model so that it can be applied on the process with minimal error.

3.2 Detection technique

For drift detection, we combine and improve on the key ideas presented in [A] and [B]. To quantify drift, we continuously compute a difference metric d_t between the probability distributions $P_{[t_1,t_2]}(\boldsymbol{Z})$ and $P_{[t_2,t]}(\boldsymbol{Z})$ as illustrated in Figure 1. $\boldsymbol{Z}$ can be any set of variables that captures the drift in the process, such as $\boldsymbol{Z} = X$ or $\boldsymbol{Z} = \{\boldsymbol{X}, Y\}$ or $\boldsymbol{Z} = \{\boldsymbol{X}|Y\}$. In this work, we use the metric Total Variation Distance [M] (or any other metric? Why).

$$d_t(Z) = \frac{1}{2}\sum_{z \in Z} |P_{[t_2,t]}(\boldsymbol{Z}) - P_{[t_1,t_2]}(\boldsymbol{Z})|$$

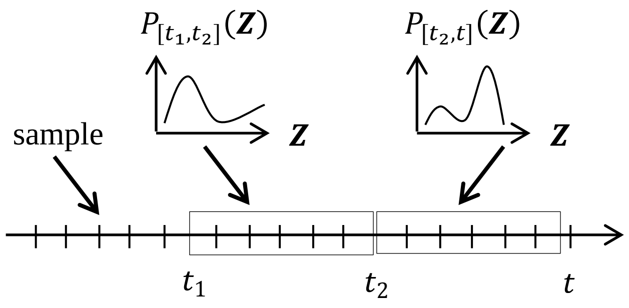
$P_{[t_1,t_2]}(\boldsymbol{Z})$
$P_{[t_2,t]}(\boldsymbol{Z})$
sample
$\boldsymbol{Z}$
$\boldsymbol{Z}$
t_1
t_2
t

Figure 1: Time windows for drift measurement

$d_t(Z)$ is tracked over time, and values above some low threshold are summed. The purpose of having the low threshold is to prevent noise or other artifacts from affecting the moving sum and triggering false alarms. When the sum increases above a certain threshold, a drift is detected and reported.

3.3 Adaptation technique

We propose an adaptation technique that employs an ensemble of weighted sub models in this work. Initially, the ensemble has only the originally learned sub model. When a drift is detected by the detection method, the following actions are taken

1. A new sub model is trained from the examples in the current window $[t_2, t]$ and added to the ensemble.
2. Weights of sub models in the ensemble are updated based on the distance between the current distribution and the distribution of the window on which the sub model was trained.

$$w_i = w_i \cdot \frac{1}{d_{[current\ window,\ i^{th}\ window]}(Z)}$$

[may have to re-work equation; eg: exponential]

Since the final model is made of the weighted sum of the sub model outputs, step 2 above ensures that sub models that were trained on sample windows similar to the current window have more weight.

[a note about the choice of Z, type of drift detected by it, ground truth, and types of models and possible ways of learning them]

[a note about how our system performs on following desirable properties

- (1) quickly adapt to concept drift
- (2) be robust to noise and distinguish it from concept drift
- (3) recognize and treat recurring contexts.

]

4. Experimental Evaluation

[to fill]

- Artificial dataset
  - Rotating hyperplane

- Real life dataset
  - Australian New South Wales Electricity Market dataset

- Gather results for following scenarios and tabulate
  - Our method
  - No adaptation (original model only)
  - Exhaustive search – look for subset that gives best accuracy on test set (lower bound)
  - Last batch/ n batches only - model trained on last batch/ n batches

4.1 Results on artificial dataset

[to fill]

4.2 Results on real life dataset

[to fill]

5. Conclusion

[to fill]

References

[A] Learning with Drift Detection – J. Gama

[B] Characterizing Concept Drift - Geoffrey I. Webb

[C] Learning under Concept Drift: an Overview - Indre Zliobaite, 2010

[D] N. Street and Y. Kim. A streaming ensemble algorithm (sea) for large-scale classification

[E] H. Wang, W. Fan, P. Yu, and J. Han. Mining concept-drifting data streams using ensemble classifiers

[F] An Ensemble Classifier for Drifting Concepts - Ralf Klinkenberg

[G] I. Koychev. Gradual forgetting for adaptation to concept drift

[H] P. Zhang, X. Zhu, and Y. Shi. Categorizing and mining concept drifting data streams

[J] Classifier Adaptation at Prediction Time - Christoph H. Lampert

[K] Continuous Manifold Based Adaptation for Evolving Visual Domains

[L] E. Levinkov and M. Fritz. Sequential Bayesian model update under structured scene prior for semantic road scenes labeling

[M] Levin D, Peres Y, Wilmer E (2008) Markov Chains and Mixing Times. American Mathematical Soc., URL https://books.google.com.au/books?id=6Cg5Nq5sSv4C

[N] The problem of concept drift: definitions and related work - Alexey Tsymbal