Identifying Predictions That Influence the Future: Detecting Performative Concept Drift in Data Streams

Brandon Gower-Winter, Georg Krempl, Sergey Dragomiretskiy, Tineke Jelsma, Arno Siebes

Utrecht University 8 Hiedelberglaan, Utrecht, 3584 CS, NL
b.gower-winter@uu.nl, g.m.krempl@uu.nl, sdragomiretsky@gmail.com, tinekejelsma@gmail.com, a.p.j.m.siebes@uu.nl

Abstract

Concept Drift has been extensively studied within the context of Stream Learning. However, it is often assumed that the deployed model's predictions play no role in the concept drift the system experiences. Closer inspection reveals that this is not always the case. Automated trading might be prone to self-fulfilling feedback loops. Likewise, malicious entities might adapt to evade detectors in the adversarial setting resulting in a self-negating feedback loop that requires the deployed models to constantly retrain. Such settings where a model may induce concept drift are called performative. In this work, we investigate this phenomenon.

Our contributions are as follows: First, we define performative drift within a stream learning setting and distinguish it from other causes of drift. We introduce a novel type of drift detection task, aimed at identifying potential performative concept drift in data streams. We propose a first such performative drift detection approach, called **C**hecker**B**oard **P**erformative **D**rift **D**etection (CB-PDD). We apply CB-PDD to both synthetic and semi-synthetic datasets that exhibit varying degrees of self-fulfilling feedback loops. Results are positive with CB-PDD showing high efficacy, low false detection rates, resilience to intrinsic drift, comparability to other drift detection techniques, and an ability to effectively detect performative drift in semi-synthetic datasets. Secondly, we highlight the role intrinsic (traditional) drift plays in obfuscating performative drift and discuss the implications of these findings as well as the limitations of CB-PDD.

Code — https://github.com/BrandonGower-Winter/CheckerboardDetection/releases/tag/v1.0

Extended version — https://arxiv.org/abs/2412.10545

Introduction

Performative Drift (PD) is a specific type of Concept Drift that occurs when predictions, made by deployed models, affect the future distributions they predict on. Prevalent in many practical domains such as Recommender Systems (Mansoury et al. 2020), adversarial settings such as malware, fraud and spam detection, learning in such scenarios is called *Performative Prediction* (Hardt and Mendler-Dünner 2023). For the most part, research has focused on settings where the presence of PD is known *a priori*. Consequently, a research gap has been formed whereby most work in the performative domain neglects the practical setting in which these performative models operate. These settings are categorised by high-volumes of heterogeneous, non-stationary and potentially transient data. Properties that appropriate themselves naturally to the domain of Stream and Online Learning (Gama et al. 2014), (Lu et al. 2018).

The primary goal of this research is to join together these two topics (Performative Prediction and Stream Learning), so that we may take existing research from one to solve problems or limitations in the other. In particular, we address the problem of detecting PD in settings where its presence is not known beforehand. At first glance, using Concept Drift detectors seems appropriate, however they are limited in that they are unable to distinguish between drift types. This means that if a traditional drift detector in used in a performative setting, one will be unable to identify if any drift detections are triggered by PD or by some other type of intrinsic drift. We address this limitation in the state-of-the-art by introducing a first-of-its-kind performative drift detector called *CheckerBoard Performative Drift Detection* (CB-PDD) and evaluate its efficacy under various experimental conditions. In short, this paper contributes the following:

1. A definition of performative drift and illustration as to why it is different to intrinsic drift.
2. A method for detecting performative drift.
3. A first-of-its-kind data generator for evaluating performative drift detectors.
4. An evaluation of CB-PDD across various experimental scenarios including a sensitivity analysis, comparison to other drift detection techniques, a robustness evaluation in settings with both performative and intrinsic drift, and a demonstration of the efficacy of CB-PDD on three datasets with imputed performative drift.

Background

Data Streams and Online Machine Learning

(Read and Žliobaitė 2023) define a data-stream as a mode of access to a potentially infinite sequence of instances generated by some concept and delivered to a learning algorithm by an instance-delivery-process (IDP). This may then be formally written as follows:

Consider a random variable X with distribution $P(X)$, a realisation (instance) of X is then denoted as $x \sim P(X)$. A stochastic generating process for X may then be written as $\{X_0, X_1, \ldots, X_n\}$ with realisations of X_i during this process forming a sequence of instances: $x_0, x_1, \ldots, x_n$.

In a supervised learning task, each instance x_i is a tuple $x_i = (x_i, y_i)$ which represents a training sample. The generation of said training sample is modelled as $(x_i, y_i) \sim P(Xi, Y_i)$ where $P(X_i, Y_i)$ is an abstraction of the concept (generating process) at some timestep i. Finally, a data-stream for some supervised learning task is defined as:

$$((x_0, y_0), (x_1, y_1), \ldots, (x_i, y_i), \ldots) \tag{1}$$

Where each training sample (x_i, y_i) is drawn from some concept $P(X_i, Y_i)$ at timestep i.

Intrinsic Concept Drift

In the context of data-streams, the term Concept Drift is used to describe perturbations in the data a stream learning algorithm receives. Concept Drift comes in many forms (Lu et al. 2018), but they all fundamentally describe a change in the data-generating process ($P(X_i, Y_i)$). If the generating process changes suddenly, it is known as abrupt drift. If it changes slightly over time, it is known as incremental or gradual drift, and if these changes are repetitive (e.g. public transportation usage patterns over workweeks compared to weekends), the drift is known as reoccurring drift. Formally, Concept Drift may be described as:

$$P(X_{i-1}, Y_{i-1}) \neq P(X_i, Y_i) \tag{2}$$

Where $P(X_i, Y_i)$ is the generating process of some data-stream at timestep i. When the generating process of a data-stream at two timesteps is not equivalent, Concept Drift has occurred. Concept Drift may also be modelled as:

$$P(X_i, Y_i) \sim D(\mathcal{P}) \tag{3}$$

Where $\mathcal{P}$ is the set of all possible joint distributions of X and Y and D is the distribution over them. Using Equation 2, we may also define where Concept Drift occurred from. It may occur in the prior probabilities of the class labels $P(Y)$, in the class-conditional feature distribution $P(X|Y)$, and in the posterior probabilities of the classes as well $P(Y|X)$. Lastly, it is possible that the incoming distribution $P(X)$ also changes without affecting $P(Y|X)$ (Gama et al. 2014).

Performative Prediction

(Perdomo et al. 2020) introduced the concept of *Performative Prediction*. Inspired by Economic Forecasting (Hardt and Mendler-Dünner 2023) and Strategic Classification (Hardt et al. 2016), Performative Prediction is the act of training a learning algorithm whose predictions affect the distribution of future instances the algorithm predicts on.

The goal of performative prediction is to evaluate the *performative risk* a model θ produces under loss function ℓ.

$$Risk(\theta, D) = \mathop{\mathbb{E}}_{Z \sim D(\theta)} \ell(Z; \theta) \tag{4}$$

where D maps model θ to the distribution of labelled instances it produces $Z = (X, Y)$. Generally, a learning algorithm will try to minimize performative risk. This is challenging for two reasons: The distribution map D is dependent on model θ and assumed to be unknown in most cases.

To solve these two issues, a procedure called *repeated risk minimization* (RRM) is introduced (See Equation 5). RRM is simply an update rule that resembles a simple retraining process found in many Stream Learning tasks (He et al. 2011).

$$\theta_{i+1} = \underset{\theta}{argmin} \mathop{\mathbb{E}}_{Z \sim D(\theta_i)} \ell(Z; \theta) \tag{5}$$

Concept Drift and Performative Prediction

The similarities between Performative Prediction and Concept Drift are apparent. Both fields are concerned with changing distributions over time as well as introducing retraining protocols to minimize the impact said distributions have on the performance of a deployed model. The key difference however, is that Performative Prediction is concerned with distribution changes that arise from the predictions made by a deployed model whereas the source of change is not necessarily relevant in Concept Drift.

(Perdomo et al. 2020) described Concept Drift as a more general problem to solve than that of Performative Prediction. In this work, we explicitly consider Performative Prediction to be specialized (subset) task of Concept Drift research. More specifically, when the distribution changes of a data-stream are induced by a predictive model(s), we call it *Performative Drift* (PD). Alternatively, when the distribution changes of a data-stream are induced by other sources, we call it *Intrinsic Drift*.

Formally describing PD as a specialized process of Concept Drift is also intuitive. Recall Equation 3 which describes a Distribution map D from which an instance generating process (concept) $P(X_i, Y_i)$ is drawn from: $P(X_i, Y_i) \sim D(\mathcal{P})$. In the performative setting future concepts are conditioned on the currently deployed model θ:

$$P(X_{i+1}, Y_{i+1}) \sim D(\mathcal{P} \,|\, \theta_i) \tag{6}$$

Note that with this formalization, the concept P is equivalent to Z used in the definition of Performative Risk (Equation 4). It is also worth noting that Equation 6 does not make assumptions about the presence of intrinsic drift. In fact, it is entirely possible that a system with performative drift may also have intrinsic concept drift.

Feedback Loops and Performative Drift

Alternatively, PD can be modelled using feedback loops (Taori and Hashimoto 2023). There are two primary types of feedback loops. The first is a self-fulfilling feedback loop:

$$P(x \,|\, y, \hat{y}, \bar{y})_i \; < \; P(\bar{X} \,|\, y, \hat{y}, \bar{y})_{i+1} \tag{7}$$

and the second-type is a self-defeating feedback loop:

$$P(x \,|\, y, \hat{y}, \bar{y})_i \; > \; P(\bar{X} \,|\, y, \hat{y}, \bar{y})_{i+1} \tag{8}$$

where $x \in X$ is an instance with label y, prediction $\hat{y}$ (from some model θ) and *target prediction* $\bar{y}$. In a self-fulfilling feedback loop, the probability of encountering instances similar to x increases if $\hat{y} = \bar{y}$. Conversely, the probability of encountering instances similar to x decreases in a self-defeating prophecy if $\hat{y} = \bar{y}$. Note that feedback loops are not induced by an instance's label, but by its target prediction which may be triggered by either a correct classification ($y = \bar{y}$) or misclassification ($y \neq \bar{y}$).

The last component of the equation is $\bar{X}$ which is defined as the set of instances that are at least ϵ close to instance x using some distance metric (e.g. Euclidean):

$$\bar{X} = \{\bar{x} \in X \texttt{ where } dist(x, \bar{x}) \leq \epsilon\} \tag{9}$$

This distinction is important because the entity that produced x, and received feedback from the predictive model, may be capable of producing a variety of future instances that may not look exactly like instance x. As we show later, this dynamic can make it easier or harder to detect PD.

CheckerBoard Detection

CheckerBoard Performative Drift Detection (CB-PDD) is a three-stage algorithm which takes inspiration from controlled experiments and A/B testing (Kohavi et al. 2009). In the first stage, incoming instances are classified according to a parameterized CheckerBoard pattern (Figure 1) instead of a predictive model. Given some data-stream of length T, a user-defined parameter f is used to split a feature into groups. These groups are equivalent to the possible labels in a classification task (e.g. in a binary classification task, each feature is split according to f and assigned either a label of 1 or 0). When an instance is received, the CheckerBoard assigns the instance as having a predicted value equivalent to the group it belongs to. After a predefined length of time τ (called trial length), the labels associated with each group are altered (e.g. in a binary classification task, a group that had incoming instances predicted as class 1, will now be predicted as class 0). It is this alternating of a group's assigned label that enables CB-PDD to perform A/B testing on the incoming instances. By forcing prediction according to the CheckerBoard, CB-PDD can check whether regions in the feature space are subject to PD. If a self-fulfilling feedback loop is present, increases in instance density should be observed in regions where the checkerboard's predictions ($\hat{y}$) match the target prediction ($\bar{y}$). In the case of a self-defeating feedback loop, the instance density will decrease instead.

After T instances, stage two begins whereby the relative density changes are calculated for each trial period (τ). This is done by creating two windows (of size w) which contain the first and last w instances from a trial period. The relative density change is then calculated per class using these windows. The density changes are then stored for statistical testing. Two values are calculated per trial per class label, The first value is the density changes that occurred when instances were predicted correctly (Group A), while the second contains the density changes where instances were predicted incorrectly (Group B). In the third stage, a statistical test (e.g. Mann-Whitney U test) is performed on Groups

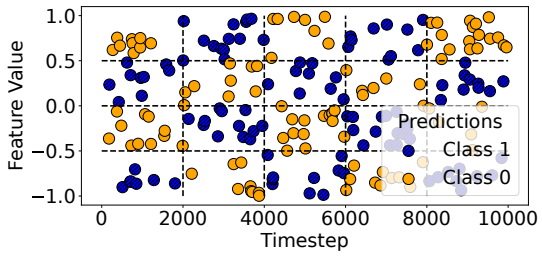

Figure 1: Example of the Checkboard Pattern produced by the Checkerboard Detector for an arbitrary feature value in a Binary Classification task. In this Figure, the CheckerBoard Detector is parameterized with $f = 0.5$ and $\tau = 2000$.

A and B using a user-defined confidence parameter α. If the p-value returned from the test is less than α, a detection event is triggered. Otherwise, no event is triggered as no performative drift was detected in the system. Note that in this work, CB-PDD is not presented as an online algorithm, this is because we only conduct a single statistical test after T instances have been evaluated. The algorithm can be altered to operate in an online fashion with sliding windows and multiple statistical evaluations (such as in KSWIN (Raab, Heusinger, and Schleif 2020)). This was deemed out of scope for this research as it introduces additional parameters to manage the sliding window and introduces the problem of repeated hypothesis testing.

Synthetic Data Generator

There is no consensus on how to model PD within a data-stream and, to the best of our knowledge, there exists no datasets with detected PD or data generators that support PD. In (Perdomo et al. 2020), they use the deployed model's parameters θ to model performative drift over time. This approach is limited in that it assumes that the user has access to θ, and is not compatible with approaches that impose alternative classification regimes such as CB-PDD.

To address this, we developed a model agnostic data generator for performative settings. Inspired by *Random Radial Basis Function (RandomRBF)* generators (Montiel et al. 2018), our generator is initialized by specifying C centroids within a feature space. Each centroid c_i is given a Gaussian distribution, a label y_i and a weight w_i. When an instance is requested, roulette wheel selection is performed (Lipowski and Lipowska 2012) using the weights. The Gaussian of the selected centroid is then sampled generating an instance x which is assigned the corresponding label y_i.

In order to simulate PD, a user is required to supply a predictor to the data generator and the intended feedback loop behaviour for each class (i.e. The target prediction $\bar{y}$ for each class and whether a self-defeating or self-fulfilling feedback loop should be used). When an instance is generated, the predictor is queried and its prediction $\hat{y}$ is stored. If $\hat{y} = \bar{y}$, the weight w_i of centroid c_i is modified. In the case of a self-fulling feedback loop, $w_i = w_i + \sigma$. In the case of a self-defeating feedback loop, $w_i = w_i - \sigma$ where σ

is a user-defined parameter that determines the performative drift strength of the system.

Experiments and Results

Using the data generator introduced in the previous Section, we evaluate CB-PDD in several experimental settings. Unless stated otherwise, $n = 50$ repetitions are conducted per parameter set evaluated and the *detection rate* is reported. $T = 100000$ instances are generated for each repetition, and a confidence value ($\alpha = 0.01$) is used for all statistical tests to determine if PD is present within a data-stream. The statistical test for this process is the Mann-Whitney U test. By default $f = 1.0$, $\tau = 1000$ and $w = 100$

For simplicity, all of these scenarios are binary classification tasks with a single feature value in which both class labels exhibit self-fulfilling feedback loops. The data generator is initialized with 100 centroids per class label (200 total) that are placed equidistantly across a feature space with range $[-1, 1]$. Each centroid is initialized with a random weight $\sim U(0, 1)$ allowing for a wide range of starting distributions to be evaluated.

Additional experiments including self-defeating feedback loops, and scenarios where only one class label is performative are included in the extended version of the paper along with a detailed parameter list for each set of experiments.

Exploration of Trial Length (τ)

The first set of experiments we conducted explored the effect trial length (τ) has on the detection rate of CB-PDD. Intuitively, as τ increases, so should the detection rate across all investigated PD strength σ values. This is due to the increased time in which a classification regime is imposed by the CheckerBoard, allotting more time for a feedback loop to develop. This theory is corroborated by our results (Figure 2), where as τ increases, so does the detection rate, regardless of σ. Conversely, our results show that if τ and σ are small, CB-PDD is unlikely to detect any performative drift. This is reflected clearly in the $\tau = 2500$ scenario where when $\sigma \leq 0.001$, we see a decline in the detection rate.

Lastly, we want to note the detection rate when $\sigma = 0.0$ (i.e no performative drift). Across all of the experiments conducted in this work, the false detection rate (i.e. when no performative drift is present but a drift detection event is still triggered) for CB-PDD remains between 0% and 20%. Furthermore, these false detections are often only raised by one of the classes and not both. These results are promising, as CB-PDD seems robust against false detections, an otherwise common issue in Concept Drift detection research.

Exploration of Feature Split (f)

The second set of experiments explored the feature split f parameter. f is required to partition a feature space into groups used by CB-PDD. Technically, any $f \in (0, l/c]$ is valid where l is the length of the feature and c is the number of classes in the classification task. Figure 3 illustrates this point where, for all f values investigated, the detection rate remains consistent across all σ investigated.

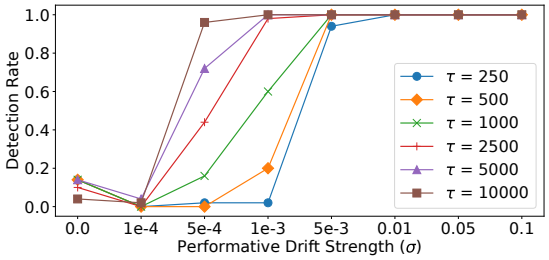

Figure 2: shows the effect trial length (τ) has on the detection rate of CB-PDD across various PD strengths (σ). The main finding is that as τ increases, so does the detection rate.

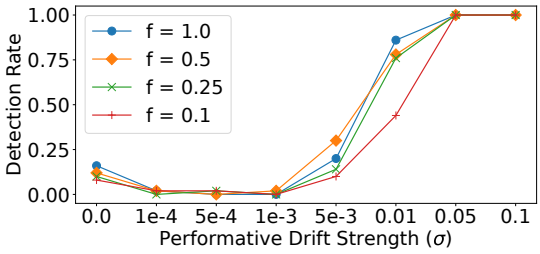

Figure 3: shows the effect f has on the detection rate of CB-PDD across various PD strengths (σ) in a low ϵ setting.

These results then seem to imply that f is not an important parameter, but that is not the case. Recall that we introduced the concept of ϵ to describe the range of future instances that might be generated by inducing a feedback loop in some instance x. If ϵ is high, future instances generated by a feedback loop induced with x will look different to x. The effect this has on f is interesting, if ϵ is high and f is low, a feedback loop induced by CB-PDD on instance x may generate future instances that are so dissimilar to x that they are assigned to a different group by CB-PDD, and thus assigned a different prediction. This has a negative effect on detection rate as inconsistent prediction assignment may reduce the momentum of a feedback loop or destroy it completely. We conducted additional experiments where we reduced the number of centroids to 10 per class label (instead of 100) and increased the spread of the Gaussians assigned to each centroid (increasing ϵ). Results are reported in Figure 4 and show this phenomena. As f decreases, the detection rate across all but the strongest PD strength settings $\sigma = \{0.05, 0.1\}$ decreases too. The takeaway from these findings are that, in general, f should be kept as high as possible to reduce the effect an unknown ϵ might have on the detection capabilities of CB-PDD.

CB-PDD with Classifiers

Traditional drift detectors typically work with a predictive model to detect drift. However, CB-PDD requires that some portion of the incoming instances be classified in accordance with the CheckerBoard predictor. Given this, we investigated

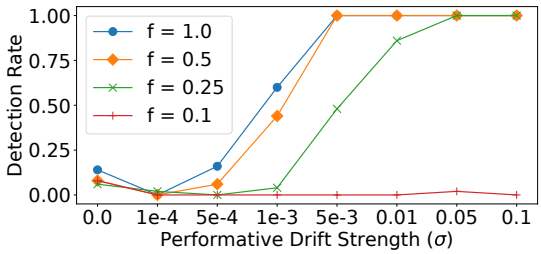

Figure 4: shows the effect f has on the detection rate of CB-PDD across various PD strengths (σ) in a high ϵ setting.

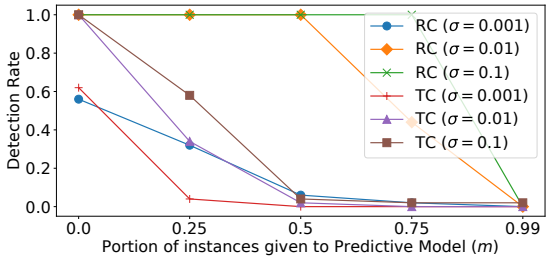

Figure 5: shows the detection rate of CB-PDD when deployed in tandem with a predictive model. m describes the portion of instances given to the predictive model. These results show that as m increases, the detection rate decreases. The trend is more noticeable in the Threshold Classifier (TC) which induces its own PD in comparison to a Random Classifier (RC) which induces no PD.

how well CB-PDD performs when paired with a predictive model. To achieve this, we ran experiments where instances are randomly assigned to either the CheckerBoard predictor or the deployed predictive model in accordance with a $mix \in [0, 1]$ parameter which describes the likelihood of an instance being given to the deployed predictor.

We investigated two predictive models: a simple Random Classifier (RC) which randomly predicts which class (1 or 0) an incoming instance belongs too, and a Threshold Classifier (TC) which predicts an instance's label in accordance with the following decision rule: 1 `if` $x > 0$ `else` 0. The RC acts as a lower-bound as it itself does not induce any performative drift, while the TC acts like an upper-bound given that it induces performative drift to a point of saturation which is akin to the performatively stable states described in (Perdomo et al. 2020). We have included a figure in the extended paper that demonstrates these effects.

In Figure 5, it can be seen that, as mix is increased, the detection rate of the CB-PDD decreases. This is due to fewer instances being classified in accordance with the CheckerBoard detector, creating weaker feedback loops which are harder to detect. This effect is less noticeable in the RC which does not induce PD, allowing up to 75% of the instances to be given to it before seeing a significant drop in detection rate. The opposite is true for the TC where a drop-off in detection rate is noticeable at $mix = 0.25$. This is due to the rate at which saturation occurs in the TC. Our results indicate that performative drift must be detected before the predictive model induces a stable state. This is because in such a stable state, PD does not actually occur as the distribution of incoming instances remains constant.

We have shown that while CB-PDD can be used with a predictive model, it works best in isolation. A potential solution to this problem is a dynamic mixing strategy whereby mix initially starts at 0.0 and is slowly increased to 1.0 over T instances. This allows CB-PDD to, initially, induce performative drift in a predictable manner which may make it easier to detect. Additional details regarding these experiments, including a method to reduce that rate at which saturation occurs, can be found in the extended paper.

Intrinsic Drift

Recall that PD is a type of Concept Drift that occurs as result of a deployed predictive model. All other types of Concept Drift are intrinsic drift. In a perfect world, performative and intrinsic drift could not coexist, but that is not the case. In order for CB-PDD to be an effective PD detector, it needs to be robust against intrinsic drift. To investigate this, we conducted experiments using two types of intrinsic drift.

Sudden Intrinsic Drift Sudden drift is caused by an abrupt change to the underlying data generating process. To simulate this in our data generator, we define a parameter E which describes the number of drift events that will occur within a simulation run. When one of these events occurs, each centroid in the data generator is assigned a new position in the feature space $\sim U(-1, 1)$.

Results are shown in Figure 6. CB-PDD achieves a low detection rate when $\sigma = 0.0$, indicating that CB-PDD is robust to sudden drift. Additionally, in both high and low intensity PD settings $\sigma = \{0.01, 0.1\}$, there is manageable deterioration in the detection rate as the number of drift events increases. However, when the frequency of drift events $< \tau$, the detection rate decreases significantly. This result is predictable as it impossible for CB-PDD to accurately determine density changes when the underlying data generating process is so chaotic. From a practical perspective, results are positive. CB-PDD appears robust against sudden intrinsic drift. However, caution should be taken when choosing an appropriate τ. Earlier results (Figure 2) show that increasing τ increases detection rate, but this will not necessarily be the case if the rate at which sudden drift events occur in the system are more frequent than τ.

Incremental Intrinsic Drift Incremental Drift occurs when the underlying data generating process changes slowly over time. To simulate this in our data generator, centroids are assigned a velocity $v_i \sim U(-1, 1)$. After an instance is generated, each centroid is moved in the feature space: $c_i = c_i + v_i * d$ where $d = 0.0001$ controls the maximum velocity of the centroids. We then utilize E which describes the number of drift events that occur within a simulation run. When one of these events occurs, the velocity of each centroid is assigned a new value $\sim U(-1, 1)$.

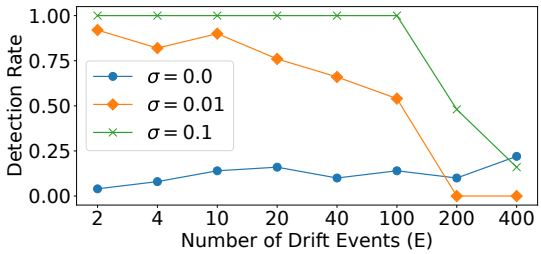

Figure 6: shows the effect sudden intrinsic drift has on the detection rate of CB-PDD across various PD strengths (σ). CB-PDD appears resilient to sudden drift except in cases where the number of drift events (E) occur at a rate greater than the trial length parameter ($\tau = 1000$).

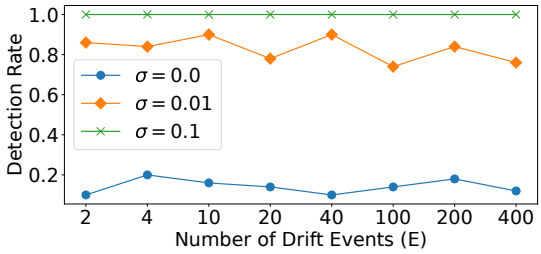

Figure 7: shows the effect incremental intrinsic drift has on the detection rate of CB-PDD across various PD strengths (σ). These results suggest that CB-PDD is resilient to incremental drift even as the number of drift events (E) increases.

Shown in Figure 7, CB-PDD is robust to Intrinsic Drift in settings when there is no PD (when $\sigma = 0.0$, the false detection rate is low). Additionally, we do not observe the same degradation in detection rate as the number of drift events increases. Results suggest that CB-PDD is immune to the effects of incremental drift. This is perhaps overly optimistic, and while CB-PDD does have resilience to incremental drift, it is quite easy to construct scenarios in which that is not the case. Consider a fail case where a cluster of instances drift from one CheckerBoard group to another before the end of a trial period. In such a scenario, CB-PDD would interpret this as a density change caused by PD. This could result in a false detection event (i.e. a false positive) if no PD is present, and cause no detection event to be triggered if PD is present (i.e. a false negative). We bring attention to this issue, because it further illustrates the importance of choosing an appropriate τ which should not overlap with these intrinsic drift phenomena. A diagram of the aforementioned fail case and additional experiments with incremental drift in scenarios with higher ϵ are included in the extended paper.

Traditional Drift Detectors

Given that PD is a type Concept Drift, it is natural to assume that traditional Concept Drift Detectors would also work in detecting PD. We test this assumption by evaluating two

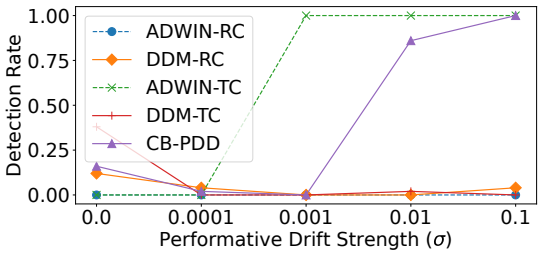

Figure 8: compares performative drift strength (σ) to the detection rate of traditional Concept Drift detectors. Results show that traditional drift detectors are reliant on the predictive model they are paired with. For example, neither ADWIN or DDM can detect PD when a Random Classifier (RC) is used, because a RC induces no PD.

popular drift detectors: ADWIN (Bifet and Gavalda 2007), a state-of-the-art featured-based detector, and DDM (Gama et al. 2004), an error-based detector. We use the implementations provided by the *river* framework (Montiel et al. 2021).

Unlike CB-PDD, both ADWIN and DDM must be used with a predictive model. We make use of the previously defined Random Classifer (RC) and Threshold Classifier (TC). Our results (Figure 8) show that neither ADWIN nor DDM are able to reliably detect PD (across any σ) when used with RC. This is because unlike CB-PDD, the other detectors don't induce their own PD, and given that the RC, doesn't induce PD either, PD is near-impossible to detect. When the TC is used, ADWIN is able to reliably detect PD at all but the lowest $\sigma = 0.0001$, outperforming CB-PDD.

DDM failed to detect PD in all scenarios investigated. DDM is an error-based drift detector. This means that a detection event is only triggered if the error of the model increases over time. In the performative setting, feedback loops don't always increase the error-rate of a predictive model, and in some cases such as these experiments, can actually decrease the error-rate. This paints an interesting picture where we can start separate the characteristics of CB-PDD from other drift detection methods. In settings with intrinsic drift and no PD, CB-PDD has shown that it will not raise false detection events. Traditional detectors will raise detection events as they are designed to detect intrinsic drift. In settings with PD and no intrinsic drift, CB-PDD can be used reliably. Feature-based detectors such as ADWIN work well too, but error-based detectors will only work if the PD causes an increase in the error-rate of the deployed model. In settings with both performative and intrinsic drift, CB-PDD has shown that it can isolate the PD and raise detection events appropriately. Other detectors can also be used in this setting, but they come with the limitation that they cannot distinguish between performative and intrinsic drift.

Semi-Synthetic Datasets

For our final Experiments, we wanted to evaluate CB-PDD on real-world datasets. Given that no such datasets exist, we take existing datasets and impute them with PD. While un-

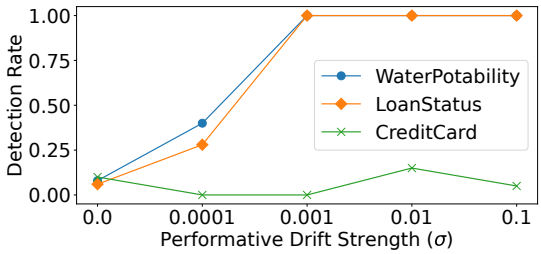

Figure 9: shows the detection rate of CB-PDD on datasets with imputed PD (σ). Results show that CB-PDD can effectively detect PD in semi-synthetic datasets, but it is susceptible to class imbalances, as shown by the *CreditCard* dataset.

desirable, this does have the benefit in that we are able to control the amount of PD each dataset exhibits. We take three datasets: Water Potability (10 features, 3276 instances) (Kadiwal 2021), Loan Status (11 features, 4269 instances) (Kai 2023) and Credit Card (29 features 284807 instances) (Dal Pozzolo et al. 2015). All three datasets are binary classification tasks with the Credit Card dataset being heavily imbalanced with only 0.172% of the instances belonging to class 1. To impute PD, we use our data generator. For each dataset, a centroid is created for each instance that only produces that instance when queried. The weights of all centroids are set to $1/N$ where N is the number of instances in the dataset. This is done to preserve the initial distribution of each dataset. From this setup, $T = 100000$ instances are sampled as in previous experiments.

Figure 9 shows positive results. For the Water Potability and Loan Status datasets, CB-PDD achieves near-perfect detection rates for all but the smallest $\sigma = 0.0001$. These results are better than our purely-synthetic experiments. We attribute this to the consistent initial distribution that each simulation starts with. In the synthetic setting, each simulation had a random starting distribution. This allowed us to investigate a wider range of scenarios than in these experiments, but it also meant that some distributions were ill-suited for CB-PDD. These results give us confidence that CB-PDD can detect PD in a practical setting. For the Credit Card dataset, CB-PDD failed to detect drift events for both classes across all scenarios. This is due to the heavy class imbalance in the dataset indicating a limitation of our method.

Limitations

Experimental results are promising. CB-PDD can detect PD in a variety of settings, is resilient to intrinsic drift, and we showed why traditional concept drift detectors are insufficient tools for isolating PD. However, this paper is not without limitations. The greatest issue is the lack of real-world data. This extends to our findings that CB-PDD's trial length (τ) parameter is sensitive to the intensity of PD a system experiences. We abstracted PD intensity (σ) and it is not clear how the values chosen in this work translate to real-world settings. We hope this paper serves as a springboard for curating datasets with known PD which CB-PDD can be evaluated on. Furthermore, we want to reiterate that CB-PDD is an intervention testing method. This means that in a practical setting, some portion of the incoming instances must be deliberately misclassified. If this is infeasible or unethical, another detection method would need to be developed.

Related Work

(Perdomo et al. 2020) pioneered Performative Prediction, but learning in the performative setting has emerged over the years under various names. (Krempl et al. 2021; Krempl, Bodnar, and Hrubos 2015) named scenarios of prediction-induced drift as Influential Machine Learning. (Khritankov 2023) contributed to formally modelling feedback loops that induce concept drift and other researchers such as (Adam et al. 2020; Taori and Hashimoto 2023) describe performative prediction scenarios as learning with feedback loops. Interestingly, their works were less concerned about finding optimal or stable points during training, but rather on the real-world implications of performative settings, such as model trust and bias amplification. Performative Prediction has also expanded since its inception. Examples include (Brown, Hod, and Kalemaj 2022; Li and Wai 2022) who have done work on state-dependent performativity while (Narang et al. 2023; Piliouras and Yu 2023) brought performative prediction to the multi-agent setting. Lastly, the notion of performativity is also well known in other related fields such as Recommender Systems (Mansoury et al. 2020) where ranking recommendations naturally influences the choices made by consumers, and Adversarial Learning (Lancewicki, Rosenberg, and Mansour 2022) (e.g. Spam Detection) where detectors must adapt to malicious entities who themselves are adapting to the detectors' predictions.

Conclusions

In this work, we investigated Performative Drift (PD), a type of Concept Drift that arises from feedback loops induced by predictions made by a predictive model. We defined PD in terms of both Performative Prediction (Perdomo et al. 2020) and traditional Concept Drift research. We then introduced *CheckerBoard Performative Drift Detector (CB-PDD)*, a first-of-its-kind PD detection algorithm. We then investigated CB-PDD across a range of scenarios using a synthetic data generator. These experiments included sensitivity analysis of CB-PDD's most important parameters, an investigation of CB-PDD's capabilities when deployed with a predictive model, an analysis of CB-PDD's resilience to intrinsic drift, and a comparison of CB-PDD to traditional drift detection algorithms. Lastly, we investigated the efficacy of CB-PDD on datasets with imputed PD.

Overall, our results are positive with CB-PDD showing high efficacy, low false detection rates, resilience to intrinsic drift, comparability to other drift detection techniques and an ability to effectively detect PD in semi-synthetic datasets. The main limitation of this work is that we only investigated CB-PDD on binary classification tasks with self-fulfilling feedback loops. Additionally, CB-PDD falters when applied to imbalanced datasets and has yet to be tested in a practical setting. Issues which we aim to address with future work.

Acknowledgements

We would like to thank the participants of the Dagstuhl Seminar 20372 "Beyond Adaptation: Understanding Distributional Changes" held in September 2020 for the valuable discussions on the topic of drift, and specifically on influential predictions, and the types and origins of drift. In particular, we would like to thank Battista Biggio at University of Cagliari, Matthias Deliano at Leibniz Institute for Neurobiology, Barbara Hammer at Bielefeld University, Eyke Hüllermeier at LMU Munich, Vincent Lemaire at Orange Labs, Michele Sebag at Université Paris Saclay, Myra Spiliopoulou at Magdeburg University, Jerzy Stefanowski at Poznan University of Technology, Dirk Tasche at the Swiss Financial Market Supervisory Authority, Oskar J. Gstrein and Andrej J. Zwitter at Rijksuniversiteit Groningen, and Mark Nelson and Margarita Quihuis from the Peace Innovation Lab at Stanford University.

References

Adam, G. A.; Chang, C.-H. K.; Haibe-Kains, B.; and Goldenberg, A. 2020. Hidden risks of machine learning applied to healthcare: unintended feedback loops between models and future data causing model degradation. In *Machine Learning for Healthcare Conference*, 710–731. PMLR.

Bifet, A.; and Gavalda, R. 2007. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM international conference on data mining*, 443–448. SIAM.

Brown, G.; Hod, S.; and Kalemaj, I. 2022. Performative prediction in a stateful world. In *International conference on artificial intelligence and statistics*, 6045–6061. PMLR.

Dal Pozzolo, A.; Caelen, O.; Johnson, R. A.; and Bontempi, G. 2015. Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE symposium series on computational intelligence*, 159–166. IEEE.

Gama, J.; Medas, P.; Castillo, G.; and Rodrigues, P. 2004. Learning with drift detection. In *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*, 286–295. Springer.

Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; and Bouchachia, A. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4): 1–37.

Hardt, M.; Megiddo, N.; Papadimitriou, C.; and Wootters, M. 2016. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, 111–122.

Hardt, M.; and Mendler-Dünner, C. 2023. Performative Prediction: Past and Future. arXiv:2310.16608.

He, H.; Chen, S.; Li, K.; and Xu, X. 2011. Incremental learning from stream data. *IEEE Transactions on Neural Networks*, 22(12): 1901–1914.

Kadiwal, A. 2021. Water Potability. https://www.kaggle.com/datasets/adityakadiwal/water-potability/data. Accessed: August 2024.

Kai. 2023. Loan Approval Prediction Dataset. https://www.kaggle.com/datasets/architsharma01/loan-approval-prediction-dataset/data. Accessed: August 2024.

Khritankov, A. 2023. Positive feedback loops lead to concept drift in machine learning systems. *Applied Intelligence*, 53(19): 22648–22666.

Kohavi, R.; Longbotham, R.; Sommerfield, D.; and Henne, R. M. 2009. Controlled experiments on the web: survey and practical guide. *Data mining and knowledge discovery*, 18: 140–181.

Krempl, G.; Bodnar, D.; and Hrubos, A. 2015. When Learning Indeed Changes the World: Diagnosing Prediction-Induced Drift. In De Bie, T.; Fromont, E.; and Leeuwen, M., eds., *Advances in Intelligent Data Analysis XIV - 14th Int. Symposium, IDA 2015, St. Etienne, France*, volume 9385 of *Lecture Notes in Computer Science*. Springer. ISBN 978-3-319-24464-8.

Krempl, G.; Hofer, V.; Webb, G.; and Hüllermeier, E. 2021. Beyond adaptation: Understanding distributional changes (dagstuhl seminar 20372). In *Dagstuhl Reports*, volume 10. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Lancewicki, T.; Rosenberg, A.; and Mansour, Y. 2022. Learning adversarial markov decision processes with delayed feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 7281–7289.

Li, Q.; and Wai, H.-T. 2022. State dependent performative prediction with stochastic approximation. In *International Conference on Artificial Intelligence and Statistics*, 3164–3186. PMLR.

Lipowski, A.; and Lipowska, D. 2012. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 391(6): 2193–2196.

Lu, J.; Liu, A.; Dong, F.; Gu, F.; Gama, J.; and Zhang, G. 2018. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12): 2346–2363.

Mansoury, M.; Abdollahpouri, H.; Pechenizkiy, M.; Mobasher, B.; and Burke, R. 2020. Feedback loop and bias amplification in recommender systems. In *Proceedings of the 29th ACM international conference on information & knowledge management*, 2145–2148.

Montiel, J.; Halford, M.; Mastelini, S. M.; Bolmier, G.; Sourty, R.; Vaysse, R.; Zouitine, A.; Gomes, H. M.; Read, J.; Abdessalem, T.; et al. 2021. River: machine learning for streaming data in Python.

Montiel, J.; Read, J.; Bifet, A.; and Abdessalem, T. 2018. Scikit-multiflow: A multi-output streaming framework. *Journal of Machine Learning Research*, 19(72): 1–5.

Narang, A.; Faulkner, E.; Drusvyatskiy, D.; Fazel, M.; and Ratliff, L. J. 2023. Multiplayer performative prediction: Learning in decision-dependent games. *Journal of Machine Learning Research*, 24(202): 1–56.

Perdomo, J.; Zrnic, T.; Mendler-Dünner, C.; and Hardt, M. 2020. Performative prediction. In *International Conference on Machine Learning*, 7599–7609. PMLR.

Piliouras, G.; and Yu, F.-Y. 2023. Multi-agent performative prediction: From global stability and optimality to chaos. In *Proceedings of the 24th ACM Conference on Economics and Computation*, 1047–1074.

Raab, C.; Heusinger, M.; and Schleif, F.-M. 2020. Reactive soft prototype computing for concept drift streams. *Neurocomputing*, 416: 340–351.

Read, J.; and Žliobaitė, I. 2023. Learning From Data Streams: An Overview and Update. *Available at SSRN 4326595:* http://dx.doi.org/10.2139/ssrn.4326595.

Taori, R.; and Hashimoto, T. 2023. Data feedback loops: Model-driven amplification of dataset biases. In *International Conference on Machine Learning*, 33883–33920. PMLR.