Analyzing and repairing concept drift adaptation in data stream classification

Ben Halstead[1] · Yun Sing Koh[1] · Patricia Riddle[1] · Russel Pears[2] · Mykola Pechenizkiy[3] · Albert Bifet[4,5] · Gustavo Olivares[6] · Guy Coulson[6]

Abstract

Data collected over time often exhibit changes in distribution, or *concept drift*, caused by changes in factors relevant to the classification task, e.g. weather conditions. Incorporating all relevant factors into the model may be able to capture these changes, however, this is usually not practical. Data stream based methods, which instead explicitly detect concept drift, have been shown to retain performance under unknown changing conditions. These methods adapt to concept drift by training a model to classify each distinct data distribution. However, we hypothesize that existing methods do not robustly handle real-world tasks, leading to adaptation errors where context is misidentified. Adaptation errors may cause a system to use a model which does not fit the current data, reducing performance. We propose a novel repair algorithm to identify and correct errors in concept drift adaptation. Evaluation on synthetic data shows that our proposed AiRStream system has higher performance than baseline methods, while is also better at capturing the dynamics of the stream. Evaluation on an air quality inference task shows AiRStream provides increased real-world performance compared to eight baseline methods. A case study shows that AiRStream is able to build a robust model of environmental conditions over this task, allowing the adaptions made to concept drift to be analysed and related to changes in weather. We discovered a strong predictive link between the adaptions made by AiRStream and changes in meteorological conditions.

Keywords Concept drift · Data stream classification · Recurring concepts

1 Introduction

Over the past few years there has been a shift towards monitoring solutions making use of multiple low-cost IOT based sensors, for example, in maintaining air quality levels. However, classification using these streams of data is not trivial. Often, there are a number of

---

✉ Ben Halstead
   bhal636@aucklanduni.ac.nz

Extended author information available on the last page of the article

Fig. 1 An example of inference.
The level of $PM_{2.5}$ at location T
at time t is predicted using the
values of neighboring sensors at
times t and $t − 1$
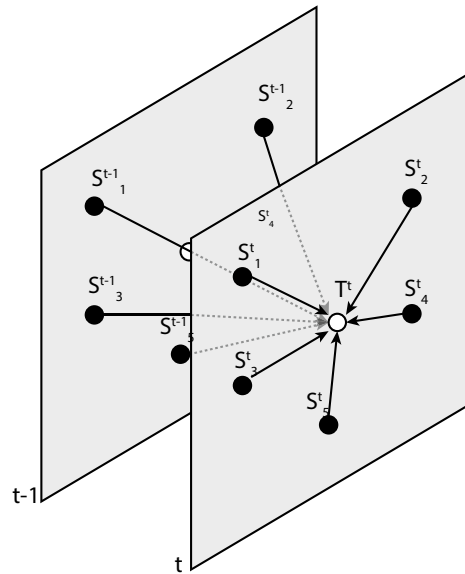
unknown factors which can influence the classification relationship between *features* and *label*. When these unknown features change over the course of a stream, it may change the distribution of data that the classifier must deal with. This problem is commonly known as *concept drift* (Gama et al., 2014). Classifiers without the ability to detect and adapt to concept drift have difficulty retaining performance as conditions change over long periods of time.

One domain which emphasizes these issues is air quality inference. An example of the inference problem we consider here is shown in Fig. 1. The current level of $PM_{2.5}$ (particles smaller than 2.5 µm in diameter) at a target location T must be predicted using the readings of neighboring sensors at times t and $t − 1$, and the level of T at $t − 1$ if it is available. Inference in air quality data is difficult due to spatial non-linearities and abrupt temporal changes. Past research (Zheng et al., 2013, 2015) has identified strong dependencies between these spatio-temporal relationships and environmental and contextual features, such as, meteorological conditions (wind), urban activity (traffic and heater use) and points of interest (locations of factories). For example, changing wind direction might change the direction in which pollution flows between sensors (illustrated by Fig. 2), or falling temperatures might encourage the use of wood burners, thus, increasing the proportion of pollution from residential areas.

These changes influence which sensor readings are most informative when making inferences. Classifiers which cannot adapt may not perform well as these changes occur. Figure 3 shows that the performance of two non-adaptive classifiers, a linear inverse distance weighted interpolator (IDW) (Wong et al., 2004) and an Ordinary Kriging (OK) classifier (Wackernagel 2013), is not stable as wind direction changes across a 12 week period of time.

A common solution to concept drift is to incorporate relevant environmental features into the model to better capture change. For example, Cheng et al. (2018) used weather type, temperature, pressure, humidity, wind, points of interest and traffic features in an attention based neural network to determine which neighboring sensors will be most
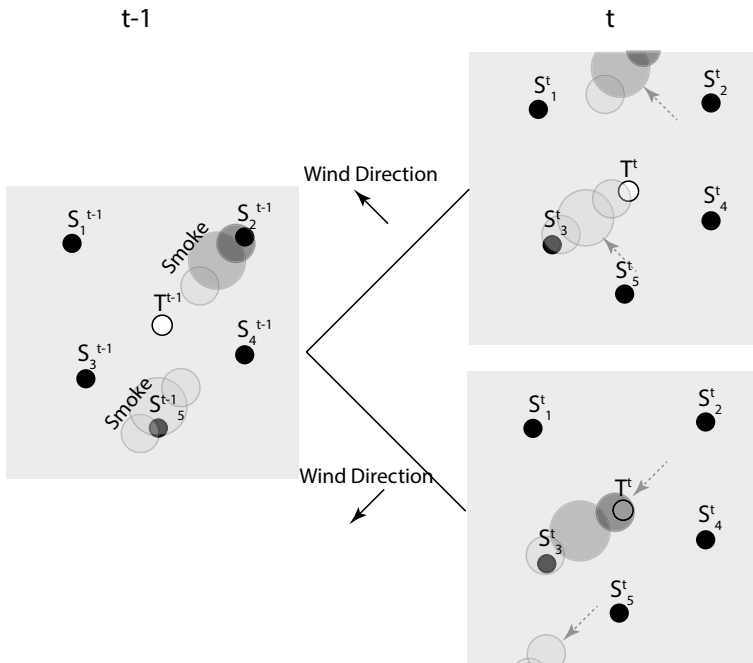
Fig. 2 An example of different spatio-temporal relationships due to wind. In the top right figure, wind from the southeast blows smoke from S_5 to T so $T^t = S_5^{t-1}$. In the bottom right figure, wind from the northeast blows smoke from S_2 to T so $T^t = S_2^{t-1}$
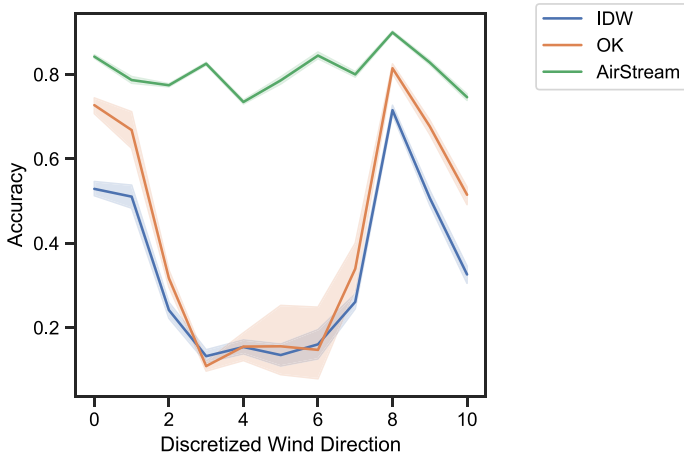

Fig. 3 Performance of linear interpolation (IDW), Ordinary Kriging (OK) and AiRStream classifiers as wind direction changes

influential at each prediction. This solution is not always practical due to the large amount of data and data sources it requires. A common challenge is a lack of good quality information for the factors relevant to concept drift. The two locations we evaluate

in this research have no reliable meteorological monitoring, making inference systems which require contextual features unusable or suffer from poor performance.

The air quality inference problem exemplifies the need for a robust classification system capable of adapting to concept drift without relying on additional data. We hypothesize that the adaptions made by such a classifier have the potential to allow a model of changing conditions to be built. This would be a valuable scientific tool in inferring the hidden factors contributing to concept drift. Recent research into data stream mining has proposed methods which are able to detect and adapt to concept drift, even in unknown factors, such as Adaptive Random Forest (ARF) (Gomes et al., 2017). However, the concept drift adaption process in these methods is not robust in noisy real-world conditions. In our experiments ARF produces many adaptions over short periods of time. While this may be acceptable from an accuracy standpoint, we show it does not produce a useful model of changing conditions.

We propose *Analysis Repair Stream*, or AiRStream, to solve this problem, a general data stream framework capable of detecting and adapting to change even in unobserved features. A novel repair algorithm allows the construction of a robust model of concept drift even in noisy real-world conditions. We show, using real data taken at two rural towns, that our system is capable of identifying changes related to wind direction and speed without those variables being available as input.

We consider observations as a stream, allowing the application of data stream mining techniques. Rather than adapting to environmental features, we adapt to changes in the distribution of streaming observations using concept drift detection methods. Reacting to these changes enables us to consider the stream as a sequence of stationary segments, allowing the application of powerful stationary classifiers. As each new segment is encountered, we build a new classifier for it or reuse an old classifier. AiRStream builds on top of this framework by introducing a repair algorithm to increase the robustness of this adaption process, allowing us to track transitions between classifiers to build a model of changing conditions. Observation noise, for example from an unreliable sensor, has the potential to disrupt the transition process by masking a change in conditions or hiding the correct classifier for a segment. Our repair algorithm allows AiRStream to detect and repair these potential errors, allowing the reuse of classifiers in the system to be a strong predictor of recurring environmental conditions. Crucially, we apply this model of changing conditions to infer meteorological information directly from the classification process.

We deployed our system on data measuring the air quality of two rural towns, Rangiora and Arrowtown. AiRStream obtained a higher predictive performance in inferring $PM_{2.5}$ levels compared to eight baseline methods. The changes captured by our system are shown to increase the ability to infer current environmental conditions above only air quality features. We present the following contributions in this paper:

- We propose a method of detecting and repairing concept drift adaption errors. By periodically sampling the accuracy of inactive classifiers, we identify cases where change was missed or misclassified. Repairing these errors increases performance and produces a more robust model of changing conditions.
- We propose a data stream based system, AiRStream, capable of adapting to changes in unknown factors. We evaluate on a real-world data set where state-of-the-art solutions cannot be used due to lack of features.
- We analyse the model of environmental conditions estimated by AiRStream. We match the model selected by AiRStream to ground truth weather conditions to investigate the

ability of AiRStream to infer current environmental conditions. We verify that changes in model are linked to true changes in weather conditions.

In the next section we present an overview of the inference problem and discuss the data sets we investigate. In Sect. 3 we outline AiRStream and in Sect. 4 we discuss a component to repair decision making errors due to noise. In Sect. 5 we discuss inferring environmental conditions using AiRStream. We evaluate AiRStream's ability to infer both $PM_{2.5}$ and environmental conditions on synthetic and real-world tasks in Sect. 6. Sect. 6.7 presents a case study analysing the relationship between adaptions made by AiRStream and meteorological conditions. In Sect. 7 we discuss related work in both air quality inference and adaptive data stream methods. Section 9 concludes with a discussion on future work.

2 Problem overview

In this section we introduce the air quality inference task investigated in this work and discuss the challenges it poses. We introduce adaptive data stream mining methods to handle these challenges.

2.1 Air quality inference datasets

To combat high rates of wood smoke pollution in rural areas, two government run studies placed ODIN wood smoke pollution sensors around rural towns. The first study, *Rangiora*, placed 13 sensors over winter, from 20 June 2017 to 25 August 2017. The second study, *Arrowtown*, placed 51 sensors between 16 July 2019 and 18 September 2019. Each sensor produced 3 readings, PM_1, $PM_{2.5}$ and PM_{10}, in minute intervals. We select $PM_{2.5}$ as the target for this work. Huggard et al. (2018) identifies $PM_{2.5}$ as having the most important impact on human health, recording the level of particulates small enough to harm human lungs, while also being the most accurately measured of the ODIN particulate matter readings.

Some sensors were only activated partway through the period, and many suffered breakages or missed readings. In order to accurately assess performance against ground truth, we select a subset of each data set with all sensors active. For *Rangiora* we select a segment of 9 sensors across 53,810 observations. For *Arrowtown* we select 10 sensors across 68,000 observations.

Each data point in the raw data consists of a timestamp, sensor serial number and numeric readings for each of the 3 measures. To preprocess the data set, we first align the timing of readings by rounding to the nearest minute. One sensor is designated a target, and its readings are discretized into 6 levels according to international $PM_{2.5}$ quality recommendations (United States Environmental Protection Agency, 2012). The $PM_{2.5}$ breakpoints of these levels are shown in Table 1. The locations of sensors in *Rangiora* and *Arrowtown* and the distribution of observed $PM_{2.5}$ levels over the evaluation period are shown in Fig. 4. We also show the locations and distribution of a similar benchmark data set from Beijing (Zhang et al., 2017). One particular challenge highlighted in Fig. 4 is class imbalance. We observe many instances of air quality level 0 but few of level 5.

The classification task is to predict the target Air Quality Index (AQI) level based on the current and previous readings of all non-target sensors and the last stable reading of the target sensor. Formally, if the sequence of readings for the sensor with index i is

Table 1 $PM_{2.5}$ Levels (based on 24 h averages)

Level	Category	Low	High
0	Good	0.0	12.0
1	Moderate	12.1	35.4
2	Unhealthy for sensitive groups	35.5	55.4
3	Unhealth	55.5	150.4
4	Very Unhealth	150.5	250.4
5	Hazardous	250.5	> 250.5

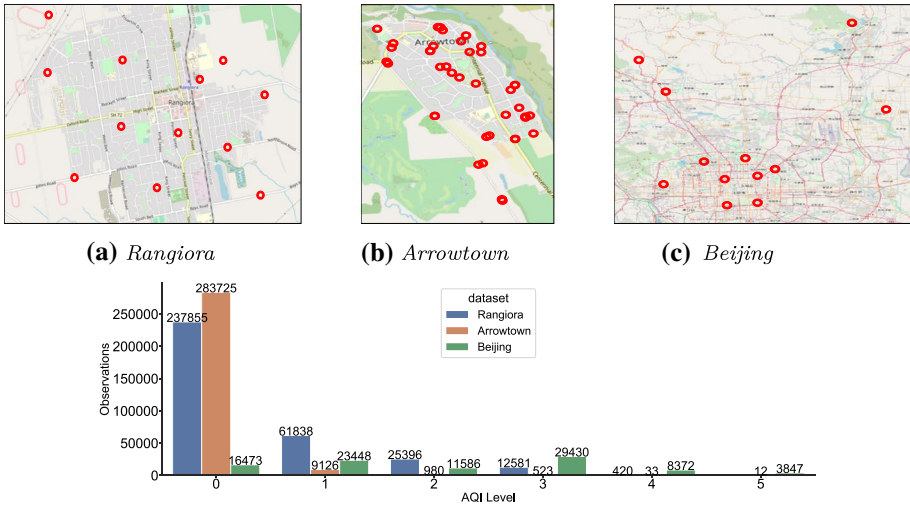**(a)** *Rangiora*   **(b)** *Arrowtown*   **(c)** *Beijing*

Fig. 4 Sensor locations and distribution of $PM_{2.5}$ levels

$S_i = (s_i^1, s_i^2, \ldots, s_i^t)$, at time t we attempt to classify the level of the target sensor l, s_l^t with features $\{s_i^t, s_i^{t-1} | i \neq l\} \cup \{s_l^{t_i}\}$, where $s_l^{t_i}$ is the most recent reading received from the target sensor. In this paper we investigate interpolation, however the task could easily be adapted to the prediction of a timestep f steps in the future by inserting s_l^t into the feature set and instead classifying s_l^{t+f}.

The goal of this classification task is to improve the performance of air quality measurement by inferring missing or unreliable readings. A secondary goal is to analyse the relationship between concept drift and environmental conditions. In many cases, determining which environmental factors cause concept drift in a dataset is a valuable outcome.

An important distinction between this task and past works is the lack of rich multi-source environmental features. These often include features such as wind speed, temperature or pollution sources like traffic density. Air pollution has complex non-linear spatial and temporal relationships which are dependent on these features. The classification task here is to not only work without these features, but to provide a signal towards what these features may be. In Sect. 7 we discuss Neural Network based methods which have achieved strong performance in similar tasks where large training sets and monitored environmental features are available, however these methods do not detect and adapt to previously unseen changes in factors not incorporated into the model, thus are unsuitable for this particular

task. Additionally, these methods do not allow us to analyse the sequence of adaptions made in response to changing conditions. The data stream approach proposed in the next section allows us to identify, model and adapt to changes in conditions without environmental features.

2.2 Adaptive classification

We consider supervised classification, in which each observation takes the form $\langle X, y \rangle$ where X is a set of features and y is a class label. The relationship between X and y may depend on a set of hidden environmental conditions, H, which are not observed by the classification system. Given two different sets of hidden conditions, $H_1 \neq H_2$, the joint probability $p(X, y|H_1)$ when H_1 is present may be different to $p(X, y|H_2)$ under H_2. It is common for H to change over time, causing $p(X, y)$ to also change over time. This is known as *concept drift* (Gama et al., 2014). For example, a change in wind direction (H) may shift which sensors (X) are upwind of the target location (y), changing spatial relationships important to the classification task. If concept drift is not dealt with correctly it may lead to poor classifier performance, for example, a classifier trained under one concept, $p(X, y|H_1)$, may achieve a lower performance under a different concept, $p(X, y|H_2)$. In Sect. 7 we discuss methods of adapting air quality inference to concept drift. A common approach is to attempt to observe more environmental factors, reducing the influence of H by shifting features into X. This approach can be costly and is not always practical.

An alternative approach is to consider the data set as a sequence of observations, known as a data stream (Gama et al., 2014). Rather than attempt to observe H directly, data stream methods (Gonçalves Jr & De Barros, 2013; Borchani et al., 2015) first identify *changes* in H, splitting a stream into segments of observations each associated with a stationary H.

Detecting a change in H is known as *concept drift detection*. Consider two time stamps T_1 and T_2, where the hidden conditions are H_1 and H_2 respectively, producing distinct joint distributions $p(X, y|H_1)$ and $p(X, y|H_2)$. The key idea in concept drift detection is to detect the *change* in distribution between two timestamps. A common method is to detect changes in the error rate of a classifier. In Sect. 7 we discuss other previously proposed methods of detecting changes in distribution. Typically, a model learns to approximate the distribution $p(X, y|H_1)$. A significant increase in the error rate of a model indicates observations are no longer drawn from $p(X, y|H_1)$, i.e., concept drift has occurred. Concept drift detection separates a stream into a sequence of stationary periods each associated with a hidden context, H, and a concept, $p(X, y)$. In order to maintain performance over the entire stream, it is common for adaptive systems to build a separate classifier for each of these stationary periods.

It is common for hidden contexts to appear in multiple stationary segments across a stream, i.e., *reoccur*, for example, seasonal contexts or those related to the business cycle. Rebuilding a classifier to represent a recurring context is inefficient. We can instead transfer learning from the last concept occurrence by reusing the classifier. The adaptive learning problem we consider in this paper could be considered to be a subset of lifelong learning (Parisi et al., 2019). This is discussed further in Sect. 7.

A common method of transferring learning stores previously used classifiers in a repository. Given a system entering the nth stationary period with hidden context H_n, if we have stored a classifier representation of each concept seen so far $M = [m_0, m_1, \ldots, m_{n-1}]$ representing hidden contexts $H = [H_0, H_1, \ldots, H_{n-1}]$ we can search M for classifiers which represent the emerging concept. If we find a stored classifier m_i which represents the emerging

distribution well, we can identify H_n as a recurrence of H_i. We can reuse m_i on the emerging segment with a lower error rate than building a new classifier. Segments featuring similar underlying conditions can be matched, building a map of how H changes across the stream. Section 7 discusses previously proposed methods for recurring concept detection. A typical approach is to test the performance of each stored classifier on a sample of observations taken from the emerging concept. A performance above some threshold is taken to be a recurrence.

A crucial characteristic provided by this approach, and useful for many real-world problems, is the ability to map changes and recurrences in H without requiring H to be observed. For example, if we identify that m_i is reused at three distinct points in a stream, it is likely that the context at these points was similar to H_i. It has been shown that data stream context can be mined for further performance and explainability benefits beyond the classification task (Gama et al., 2004; Borchani et al., 2015). In Sect. 6.7 we show that model reuse in AiRStream is related to the hidden context features wind speed and direction in the *Rangiora* dataset. In Sect. 7 we discuss previous works which make use of concept drift detection and model reuse. The next section describes how AiRStream applies these methods to noisy real-world streams.
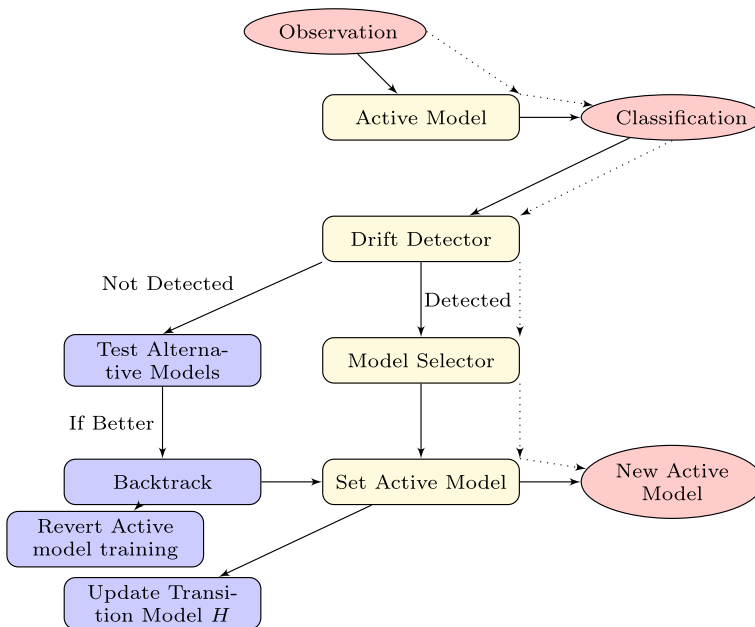

Fig. 5 AiRStream System Overview. Dotted lines indicate a basic data stream framework flow, solid lines indicate AiRStreams flow, including our additional components in blue

3 Inference system

We first give an overview of AiRStream, shown in Fig. 5 before describing in detail our proposed components.

We will briefly introduce two important components of data stream classification, incremental learners and drift detectors. An incrementally learning classifier trains on observations sequentially. Unlike a batch trained model, an incremental learner can make predictions anytime. This allows rapid adaption to change by creating a new classifier. The incremental learner we use here is the Hoeffding Tree (Gama et al., 2003), an online decision tree method used in state of the art methods (Gomes et al., 2017). The Hoeffding Tree was chosen as it provides strong performance guarantees (Losing et al., 2018), and provides a representation of a concept that can only grow. This provides a more robust representation of a concept, allowing tracking over the course of the stream.

We detect change in the distribution of data using an error rate drift detector. Concept drift is detected as a statistically significant change in the error rate of a stationary classifier as observations are processed. We use the ADWIN drift detector (Bifet & Gavalda, 2007), a top performing dynamic windowing based technique. We empirically justify this choice in Sect. 6.

These two components are the basis of the data stream framework shown in Figs. 6 and 7. A Hoeffding tree is initialized as the *active model*. The active model is incrementally trained on incoming observations with its error rate fed into the drift detector. When a drift is signalled by the detector, the active model is deactivated and added to a set of inactive models. A *model selection process* is initiated to select a new classifier best suited to the new stationary stream segment. The model selection process compares the performance of all inactive models, as well as a newly built model, over a recent window. This process attempts to pick the model most suitable for the emerging segment. The best performing model is selected as the new active model. This process repeats for each new segment in the stream. Figure 6 shows an example of a stream with three segments. A drift is signalled at time t_{n_1}, and the active model transitions to a new model, Model 2. Model 2 is used to classify observations until the next concept drift is detected at t_{n_2}. The model selection process matches the original Model 1 as the most suitable model.
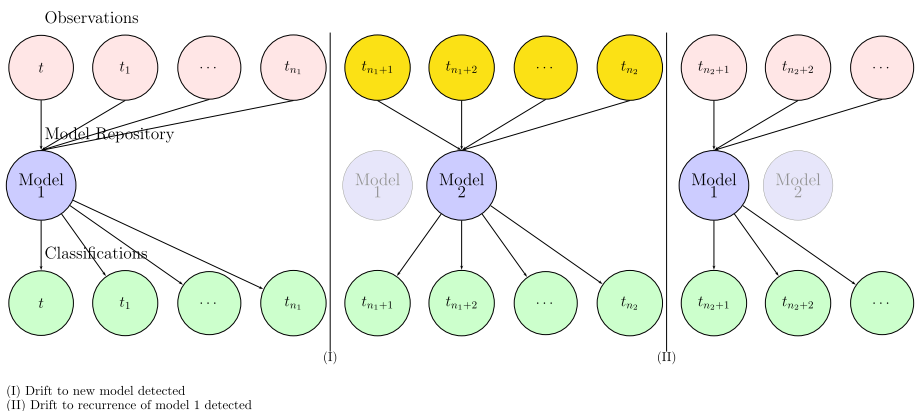

(I) Drift to new model detected
(II) Drift to recurrence of model 1 detected

Fig. 6 Basic data stream framework example timeline showing a transition to a new model and a transition to a recurring model
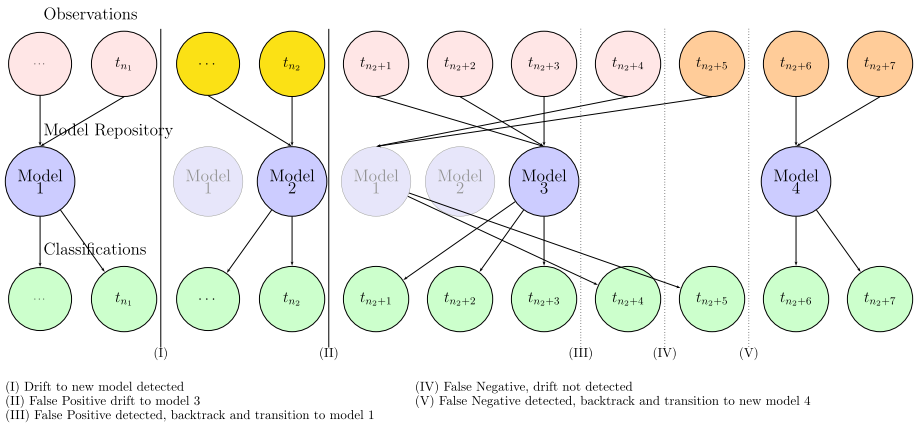
Fig. 7 Proposed system example timeline, including the repair of a false positive error and a false negative error

We propose additions to this basic framework, shown in blue in Fig. 5. Intuitively, if we chose to reuse the same model on a different segment of the stream this indicates that the segments displayed a similar relationship between features and label. We hypothesize that a similar relationship between features and label indicates similar environmental conditions. A model can then be considered as a representation of some set of environmental conditions. By building and maintaining a history of active models, we can infer environmental changes based on the transitions made during this history. We define the *Model History*, M_H, of our system as the sequence of active models used to classify the observations in a data set, $M_H = (A^1, A^2, \ldots, A^t)$, where A^t is an identifier of the active model used to classify the observation at time t.

We treat the feature values taken as input by AiRStream as potential sources of noise. To keep models as clean representations of environmental conditions given the noisy readings generated by potentially unreliable data, we implement a method to repair sub-optimal transitions. We detail this component in the following section.

4 Drift repair

Concept drift adaption can be thought of as dividing a stream into stationary segments and assigning a model to classify observations in each segment. We can consider a stream as a sequence of stationary segments $S = \{S_1, S_2, \ldots, S_\infty\}$. Each segment S_t at time t is associated with a hidden context H_t from the sequence $H = \{H_1, H_2, \ldots, H_\infty\}$ and data distribution $p(X, y|H_t)$. Given a set of models in a repository, the goal of a recurrent concept system is to identify for each segment S_t the optimal model M_{ot} which best represents $p(X, y|H_t)$. In a robust system, if we observe that a model is active at time t, we can assume it is the optimal model, M_{ot}. We can then estimate that the current hidden context is H_t. The estimated context at each time step can be analysed to learn about changes and recurrences in environmental conditions. This analysis is discussed further in Sect. 5.

Given the current active model of a system as M_{at}, we define cases where a suboptimal model is active, $M_{at} \neq M_{ot}$, as *adaptation errors*. Adaptation errors have three main negative effects.

- Current classification performance is reduced. Error rate is increased due to the use of a classifier misaligned with the true distribution of data.
- Future classification performance is reduced. If M_{at} is the optimal model for a future segment S_f, its performance on S_f will degrade as it is trained on observations drawn from S_t which display a different relationship.
- The relationship between model use and data stream context is reduced. If we do not observe M_{ot} and instead observe M_{at}, the context estimated by the system is less associated with the true context H. This limits our ability to analyse changes in environmental conditions.

Adaptation errors may be caused by many factors. In real-world systems, for example, noise can often lead to errors in concept drift detection or model selection. Feature imbalance, as identified in Fig. 4, can also cause adaptation errors. For example, changes in the distribution of a minority class may shift M_{ot} without being easily detected by concept drift. In this work, we propose a method to identify and repair adaptation errors. We identify two main cases of adaptation errors causing $M_{at} \neq M_{ot}$.

1. False Positive errors occur when the transition from S_{t-1} to S_t is correctly detected, but the active model selected is not M_{ot}. This is shown in Fig. 7 at point (II). As model selection is based on performance over a window of observations, noise in this window can make it difficult to determine the most suitable model to use.
2. False Negative errors occur when the transition from S_{t-1} to S_t is not correctly identified, and $M_{at} = M_{a(t-1)}$ rather than M_{ot}. This can occur when fluctuations in error rate due to noise mask an error rate change due to concept drift. This is shown in Fig. 7 at point (IV).

We propose a drift repair mechanism to mitigate these cases. We make the observation that in all cases of adaptation errors, there is some optimal model M_{ot} which better represents the distribution of data seen in incoming observations. Testing M_{ot} on a window of current observations would show higher classification performance than M_{at}. This forms the basis of our drift repair component. We identify false positive and false negative errors by periodically testing inactive models against the active model. If we identify that a tested model has a higher performance than M_{at}, we can conclude that $M_{at} \neq M_{ot}$ and identify the new model as M_{ot}.

When a new M_{ot} is identified, we initiate a *backtrack* repair. The negative effect on classification performance cannot be reversed for classifications already made, however we can limit the other impacts of adaptation error. In order to minimize future classification error we set the active model to M_{ot} for the remainder of S_t. We can also repair the negative effect on training and model history. Future classification performance is maintained by reverting M_{at} to a saved backup, or *restore point*, to undo training done on observations drawn from S_t. The relationship between model use and data stream context is maintained by replacing instances of M_{at} in our model history with M_{ot}.

Algorithm 1: Drift Repair

Input: M, the set of all inactive models, M_{at}, the current active model, t, the
current time step, t^r, the time step the last restore point was set

Data: a_b, the active model backup created at the last restore point

if $t - t^r \geq R_p$ **then**
$\quad$ $a_b \leftarrow M_{at}$, $t^r \leftarrow t$, $K \leftarrow$ ModelSelection(M)
$\quad$ Initialize k_a to be the running κ statistic and σ_a to be the running standard
$\quad$ $\quad$ deviation of k_a
$\quad$ **for** $k \in K$ **do**
$\quad$ $\quad$ Initialize k_k to be the running κ statistic, σ_k to be the running standard
$\quad$ $\quad$ $\quad$ deviation of k_k, p_k to be 0 and d_k to be a drift detector

if $t - t^r \leq R_l$ **then**
$\quad$ $k_a, \sigma_a \leftarrow$ GetPerformance(t)
$\quad$ **for** $k \in K$ **do**
$\quad$ $\quad$ $k_k, \sigma_k \leftarrow$ GetPerformance(t)
$\quad$ $\quad$ p_k += OneTailTTest($k_k - k_a, \sigma_a, \sigma_k$)
$\quad$ $\quad$ AddObservation($d_k, k_k - k_a$)
$\quad$ $\quad$ **if** $p_k/(t - t^r) \leq 0.05$ **then**
$\quad$ $\quad$ $\quad$ $M_{at} \leftarrow a_b$, SignalBacktrack(k)
$\quad$ $\quad$ **if** CheckDrift(d_k) *and* $k_k > k_a$ **then**
$\quad$ $\quad$ $\quad$ $M_{at} \leftarrow a_b$, SignalBacktrack(k)

The proposed algorithm for identifying and repairing cases where $M_{at} \neq M_{ot}$ is shown in Algorithm 1. After each transition and periodically every R_p observations an *error detection period* is started, testing the following R_l observations for adaptation error. We place a *restore point* at the start of each error detection period, creating a_b as a backup of the active model M_{at}. We then perform a model selection step to select the K best performing inactive models on a recent window as alternative models to test. Over the course of the error detection period we calculate the running κ statistic κ_a for the current active model and each of the alternatives $k \in K$ as κ_k.

A confidence threshold is calculated to determine if any alternative models are a better fit for the current stream. This confidence threshold is made up of two parts, a statistical confidence test which captures a gradual change in performance and a change detection scheme which captures a quick change in performance. Gradual change confidence is determined by a one-sided t-test, testing the null hypothesis that κ_k is less than or equal to κ_a. If the average p-value over the testing period falls below a threshold, i.e., 0.05, we determine $k = M_{ot} \neq a$ and a *backtrack* repair is initiated. To capture quick changes in model performance, we run the ADWIN change detection method on the performance difference $\kappa_k - \kappa_a$. A detection of change in this stream while $\kappa_k - \kappa_a$ is positive determines $k = M_{ot} \neq a$ and a *backtrack* repair is initiated. This quick change could come from a change in distribution missed by concept drift detection.

When a *backtrack* repair is initiated, we deactivate the active model and revert it to its state at the latest restore point. If the latest restore point was at a transition we classify the error as a False Positive error, i.e., the transition was to the wrong model. If the latest restore point was not at a transition, we classify the error as a False Negative error, i.e., a drift occurred and was not detected, leaving the system using a sub-optimal model. In both cases model k is activated and the system continues in its new state.

4.1 Time and memory complexity

Incremental learners and drift detectors such as Hoeffding Tree and ADWIN used in AiR-Stream, have a constant time complexity per observation. The time complexity of the base data stream framework without model selection or repair is then $O(1)$ per observation.

Given a repository of size $|M|$, the model selection process runs each model in the repository on a window of size $|w|$ to calculate a performance measure. The time complexity of the model selection step is $O(|M||w|)$. Given the per observation probability of concept drift as d, the time complexity of the base framework including model selection is $O(d|M||w|)$ per observation. Our proposed repair method additionally runs model selection periodically every R_p observations. The probability of an additional model selection event is $\frac{1}{R_p}$ per observation. This gives a total model selection time complexity of $O\left(\left(d + \frac{1}{R_p}\right)(|M||w|)\right)$ per observation.

During the error detection period, K alternative models are run alongside the active model. This occurs for R_l observations out of every R_p observations. This means we run $(K + 1)$ models for a proportion $\frac{R_l}{R_p}$ of the stream, and only the active model for the remaining $1 - \frac{R_l}{R_p}$ proportion. This gives a time complexity of $O\left((K + 1)\left(\frac{R_l}{R_p}\right) + \left(1 - \frac{R_l}{R_p}\right)\right)$ per observation. AiRStream's overall running time per observation is:

$$O\left[\left((K + 1)\left(\frac{R_l}{R_p}\right) + \left(1 - \frac{R_l}{R_p}\right)\right) + \left(d + \frac{1}{R_p}\right)(|M||w|)\right]$$

We compare this to an ensemble classifier, which runs $K + 1$ models for all observations, $O((K + 1))$ per observation. Ensemble systems which adapt to recurring concepts using model selection have a time complexity of $O((K + 1) + d|M||w|)$ per observation. Our method is able to run only one model for a $1 - \frac{R_l}{R_p}$ proportion of the stream, offering a significant time complexity benefit over the ensemble case.

In terms of memory complexity, if γ is the size of a model we require $O(\gamma(|M| + 1))$ space to store the repository and the restore point. This is equivalent to storing one additional model in either an ensemble or any system which stores a repository of past models. In Sect. 6, we empirically evaluate both runtime and memory use.

5 Condition inference

When AiRStream detects a concept drift, the active model is changed to one that matches recent observations. We hypothesize that a major driver of these concept drifts is change in environmental conditions. This hypothesis indicates that the transitions between active models in AiRStream are linked to changes in environmental conditions, and further, that when a model is reused similar environmental conditions are present. Under this hypothesis, matching weather conditions to AiRStream active models may allow environmental conditions to be inferred in the future. For example, consider a scenario where meteorological data was temporarily recorded in a location where AiRStream was active. If the post-hoc analysis revealed a strong relationship between a given set of weather conditions and the use of a given model, we may infer the similar weather conditions are present the next time the active model is used. We consider two methods for relating a set of environmental

conditions to the use of an active model, a recall and precision based method and a classifier based method.

We denote the *Condition History*, C_e, as the sequence of discretized observations of an individual environmental source e over the time period of a particular data set, $C_e = (E^1, E^2, \ldots, E^t)$, where E^t is the level of e at time t.

We evaluate the relationship between model use and environmental conditions by matching patterns in M_H and C_e. We consider the precision, $P(a, l)$, and recall, $R(a, l)$, obtained by matching each environment level l to the use of a particular active model a.

$$R(a, l) = \frac{|\{t | A^t = a, E^t = l\}|}{|\{t | E^t = l\}|} \quad P(a, l) = \frac{|\{\{t | A^t = a, E^t = l\}|}{|\{t | A^t = a\}|}$$

$$F1_c(a, l) = 2 \frac{R(a, l)P(a, l)}{R(a, l) + P(a, l)}.$$

In this case, precision measures the strength of the relationship 'model a is active therefore e has the value l', while recall measures the proportion of observations where $e = l$ where this relationship holds. The $F1_c$ score combining recall and precision measures the overall strength of the relationship. We calculate an overall score of relationship, C-F1, as the average $F1_c$ score obtained by matching each level l to the model a maximizing $F1_c(a, l)$:

$$\text{C-F1} = \mathbb{E}_a \left[\max_l F1_c(a, l) \right]$$

We also consider the ability to train a secondary machine learning model to predict the level of e given the current active model. We train a classifier using M_H as the set of features and C_e as the set of labels. The predictive ability of this classifier indicates the strength of the relationship between M_h and C_e.

We investigate the strength of the relationship between the active model used by AiRStream and environmental conditions in the next section, and find evidence that both wind speed and direction can be inferred through these methods in the *Rangiora* data set.

6 Evaluations

In this work we present AiRStream, a classification framework able to robustly adapt to concept drift using a novel repair component. We are interested in two performance measures, classification performance and the relationship created between model use and data stream context. In this section we answer three research questions: (1) How does the performance of AiRStream and its repair component compare to state-of-the-art methods? (2) Does this performance translate to a real-world task, namely air quality inference? and (3), Can the hidden underlying conditions of a data stream be inferred from the model use of a recurring concept system?

We first compare the performance of AiRStream and its repair component to eight baseline methods using synthetic datasets with known dynamic conditions. We then test the accuracy of the $PM_{2.5}$ levels inferred by each system on three real-world air quality tasks. We finally present a case study investigating the link between changes detected by our system and changes in environmental conditions in real-world data streams. All code and additional significance analysis can be found at https://bit.ly/2YnFe1p.

6.1 Dataset setup

We evaluate the performance of AiRStream using both synthetic and real-world datasets. Synthetic datasets allow for the changes in conditions creating concept drift to be isolated and recorded, while real-world datasets test performance in practical tasks. We use two synthetic data sets created using the Random Tree (TREE) and Radial Basis Function (RBF) generators available in MOA (Bifet et al., 2010). The difficulty of each generator was tuned such that a model takes approximately 4000 observations to train. This describes concepts for which reuse is useful, i.e., not trivial to learn. For the TREE generator we set the maximum tree depth to 2, minimum tree depth to 1, number of classes to 2 and use 3 categorical and 4 numeric features. For the RBF generator we set the number of centroids to 8, number of classes to 4 and number of features to 7.

In order to test real-world performance, we apply AiRStream to infer $PM_{2.5}$ levels in two rural towns given the presence of missing labels, as described in Sect. 2.1. We also evaluate on a similar data set of readings taken in Beijing (Zhang et al., 2017). There are three main differences in this data set compared to the other data sets. Firstly, it is an urban environment compared to small rural towns. Secondly, the sensors are spread much further apart (across the Beijing metro area), and lastly, all readings are recorded every hour compared to every minute. For all data sets, at each observation a classifier has access to the current and previous numeric readings of surrounding sensors, as well as the last seen $PM_{2.5}$ level of a target sensor. The task is to infer the current $PM_{2.5}$ level of the target sensor.

To allow evaluation against ground truth, we select a portion of the data set where all labels are available and simulate missing target sensor readings. We select b observations, drawn uniformly from the selected portion, as 'broken' and mask the readings from the target sensor in these observations. Most sensor breakages in the data set last for longer than one minute, thus appear as blocks of missing readings rather than single observations. To recreate this effect we select the b_{period} readings following the initial b breakages to also be masked. When a label is masked, it is not available in the feature set of the next observation, and cannot be trained on. We set b as 3% of the size of each data set with $b_{period} = 60$. All experiments are repeated 10 times on all possible target sensors, with the average results being reported.

6.2 Baseline comparisons

We compare our system to a range of eight baseline methods using different approaches to classification.

1. *Chance and No-Change Classifiers (NC)* These simple methods predict a random label drawn from the distribution of a given classifier, and the most recent non-masked label, respectively. The No-Change classifier has been shown to be very effective in many data stream classification tasks affected by auto-correlation (Žliobaité et al., 2015). We compare to the chance classifier by using a κ statistic measure (Žliobaité et al., 2015) in our evaluation.
2. *Inverse Distance Weighted interpolation (IDW) and Gaussian interpolation (NORM)* These methods infer the current target $PM_{2.5}$ level by interpolating current readings from surrounding sensors. IDW uses the average reading weighted by inverse distance to the target, while Gaussian interpolation assumes pollution reported by each sensor falls off

as modeled by the Gaussian distribution $X \sim \mathcal{N}(0, \sigma^2)$. As in U-Air (Zheng et al., 2013) we set σ to be the average distance between any two sensors. We use the RBF function in SciPy to implement NORM.

3. *Ordinary Kriging (OK) and Random Forest (RF)* To test against baselines taking into account spatial and temporal features, we implement ordinary Kriging with a linear kernel and a non-streaming random forest method (RF). We treat RF as a baseline batch non-adaptive system. In this case, RF is trained on a large training set then receives no further training. This simulates a scenario where a batch method is deployed and cannot be taken down to be retrained over the deployment period. This baseline measures the value of online training for adaptation. We note that the online methods observe more total training data so performance cannot be directly compared. Ordinary Kriging is an advanced spatial interpolation method incorporating feature covariation, while random forest is a tree based method similar to the Hoeffding trees in our system. We use the GaussianProcessClassifier in SKLearn with default parameters to implement OK. We use the Scikit-Multiflow implementation with default parameters for RF.

4. *Adaptive Random Forest (ARF)* A state of the art data stream method, Adaptive Random Forest (ARF) (Gomes et al., 2017). ARF uses an ensemble of Hoeffding Tree classifiers and can detect and adapt to concept drift, however it does not consider reusing classifiers on multiple stream segments. We use the Scikit-Multiflow implementation with default parameters for ARF.

5. *Dynamic Selection Based Drift Handler (DYNSE)* DYNSE (De Almeida et al., 2016) is a streaming ensemble system which uses dynamic classification to select ensemble members from a repository. Dynamic classification allows DYNSE to adapt to new concepts as well as recurring concepts. We use a DYNSE classifier implemented in MOA. We use default parameters recommended by the authors, including the KNORA-E classification engine (Cruz et al., 2018).

To evaluate our proposed repair component, we compare two versions of our AiRStream system. A base version not containing the repair component and a full version are denoted as AS_b and AS_r respectively, in the following results.

6.3 AiRStream parameters

The recurrent concept base framework requires selection of two components, an incremental classifier and a drift detector. It also introduces a window size parameter for controlling the number of recent observations used for model selection, and a maximum repository size $|M|$. AiRStream introduces 3 additional parameters: repair sensitivity—the threshold for triggering a repair, repair period—the number of observations between each periodic drift repair test, and the number of inactive models tested at each drift repair step, denoted k. In the next section we discuss how components are selected in AiRStream. In the following section we discuss how parameter values are selected.

6.3.1 Component selection

As in other methods, both the incremental classifier and drift detector used in AiRStream are interchangeable. We choose the Hoeffding Tree classifier as the base incremental classifier as it provides strong performance and behaviour guarantees (Losing et al., 2018).

In all experiments, the active incremental classifier is trained prequentially, i.e., evaluation measures are calculated on the incoming observation and then the model is updated to incorporate the new observation. In order to select a drift detector, we note that our drift repair mechanism depends on the characteristics of drift detection. For example, a highly sensitive drift detector may cause many false positive errors with few false negative errors, while a low sensitivity detector may display the opposite. This means that the repair component may provide more benefit under different drift detection schemes.

We perform an experiment comparing AiRStream with repair to a variant without repair across five popular drift detection methods. All other parameters are held constant. We compare ADWIN, Drift Detection Method (DDM) (Gama et al., 2004), Early Drift Detection Method (EDDM) (Baena-Garcıa et al., 2006), and Hoeffding Drift Detection Method (HDDM) (Frias-Blanco et al., 2014), HDDM_A and HDDM_W. Table 2 shows the κ statistic (Žliobaité et al., 2015) and C-F1 of each AiRStream variant using each drift detection method on two synthetic datasets. Synthetic streams are constructed by alternating 3 concepts using both abrupt and gradual drift over 50,000 observations. We observe that in all cases, the repair component gives higher κ statistic and C-F1. This indicates that repair works over a range of drift detection methods. We also see that using DDM and EDDM gives a large difference in performance between AS_r and AS_b, while using ADWIN or HDDM_W gives a relatively small difference. EDDM is designed to be sensitive to drift, which may create more adaption errors to be repaired, leading to the increased benefit of the repair component. ADWIN achieves the highest overall performance and C-F1, so we select ADWIN as a high performing drift detection method to use in the following experiments.

6.3.2 Parameter robustness and sensitivity analysis

We tested performance across four parameters, window size, repair sensitivity, repair period and k. For analysis, we remove repository size, $|M|$, as a parameter by selecting a repository size larger than required by any dataset. This simplifies evaluation by removing the effect of memory management, which is investigated in other work (Chiu & Minku, 2018). Figure 8 shows the sensitivity of our system to parameters on the *Rangiora* data set, with standard deviation given by 10 repetitions. We observe that only window size has a large impact on performance. A window size which is too small does not give model selection enough observations to be robust. A window size which is too large contains observations prior to the concept drift, disrupting model selection. We find that both extremes negatively impact classification performance. Increasing repair sensitivity improves classification performance, indicating that the repair component positively impacts performance. Performance is consistent at different levels of the repair period parameter. We set the length of each repair testing period to be a fixed ratio of the repair period. Under a small repair period, we test models for shorter periods of time which reduces sensitivity in detecting adaption errors. However, a smaller period also tests more often which may increase sensitivity. Similarly, a long repair period tests models over a longer period of time but tests less often. This trade-off may produce the consistent levels of performance. Performance is also consistent under changing k. This is expected, as models are ranked according to suitability in the model selection stage. Additional models added by a higher k are less likely to be selected by the repair component, so largely do not contribute to performance. A consistent level of performance here indicates that model selection is performing well when ranking models. A smaller repair period and larger k increase runtime.

Table 2 Classification and C-F1 performance of repair across drift detection methods

| Drift detector | Radial basis function | | | |
| | κ statistic | | C-F1 | |
	AS_r	AS_b	AS_r	AS_b
ADWIN	0.65 (0.04)	0.64 (0.04)	0.59 (0.06)	0.54 (0.05)
DDM	0.63 (0.04)	0.51 (0.05)	0.50 (0.06)	0.31 (0.04)
EDDM	0.60 (0.03)	0.51 (0.04)	0.46 (0.05)	0.31 (0.04)
HDDM_A	0.63 (0.04)	0.57 (0.04)	0.51 (0.05)	0.42 (0.05)
HDDM_W	0.64 (0.04)	0.62 (0.04)	0.53 (0.04)	0.48 (0.04)
	Random tree			
ADWIN	0.63 (0.06)	0.56 (0.08)	0.64 (0.06)	0.55 (0.07)
DDM	0.53 (0.08)	0.45 (0.05)	0.49 (0.06)	0.30 (0.05)
EDDM	0.49 (0.06)	0.45 (0.05)	0.43 (0.05)	0.28 (0.01)
HDDM_A	0.50 (0.07)	0.45 (0.05)	0.44 (0.04)	0.39 (0.04)
HDDM_W	0.54 (0.06)	0.49 (0.06)	0.48 (0.07)	0.47 (0.04)

Values in parenthesis show standard deviation

For all following experiments we use a window size of 1500, a repair sensitivity of 0.2, a repair period of 2000 and k of 5. We find empirically that this is a good trade-off between performance and runtime.
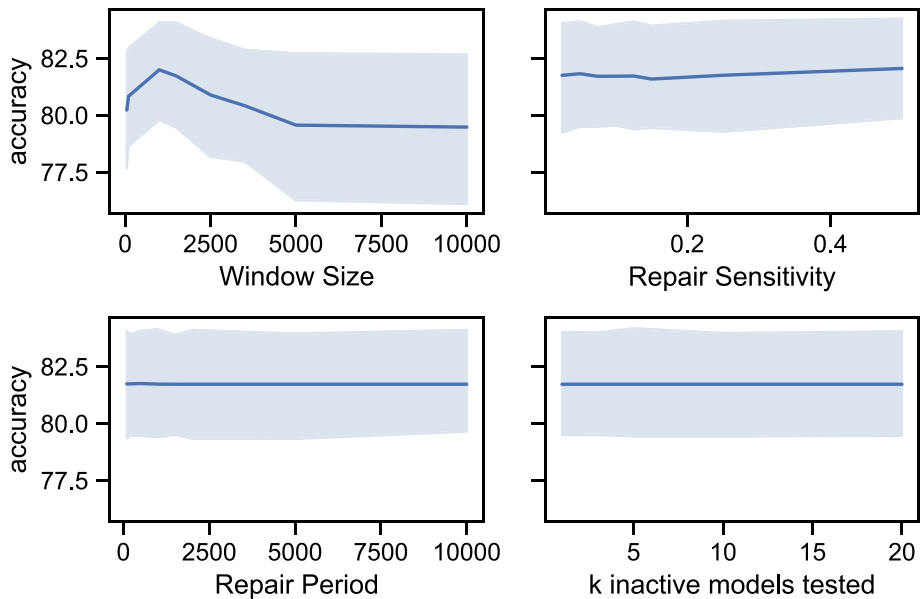

Fig. 8 Sensitivity analysis based on parameter tuning

6.4 Synthetic results

In this experiment we answer research question (1): how does the performance of AiR-Stream and its repair component compare to state-of-the-art methods? We investigate whether AiRStream provides higher classification and context modeling performance when adapting to concept drift than alternate methods on synthetic data. The aim of this experiment is to measure drift adaption performance, rather than how well each method learns the synthetic function. For each test, 20,000 observations were drawn alternating between two concepts with abrupt and gradual concept drift with a width of 4000 observations. Using synthetic data allows the 'environmental conditions' driving concept drift, in the synthetic case the generating function acting at each concept, to be known and isolated. Known contexts allow us to quantify the relationship between environmental conditions and model use for each system using the C-F1 measure. We also test the impact of the repair component by comparing the AiRStream variant without repair, AS_b, to the system with repair, AS_r. The task in this experiment is not interpolation so we do not compare to the IDW and NORM interpolation baselines.

6.4.1 Classification performance

Table 3 shows the performance of each method on synthetic data sets generated using the RBF and TREE generators. Results shown are averaged over 100 runs. Each run uses a different set of synthetic concepts. The initial 20% of each stream is used for training. For streaming systems we report prequential performance measures on the remaining observations. The RF method does not train further, so report holdout performance measures. To measure classification performance invariant to the length of each concept, we report the κ statistic (Žliobaité et al., 2015) for the 150 observations after each drift (κ_{150}). A higher performance in this period indicates better adaption to drift, which is similar to the idea of recovery analysis (Shaker & Hüllermeier, 2015). Note that C-F1 shown in Table 3 is not a measure of performance, rather it measures condition inference. To test for significance in Table 3 as well as the following Tables 4, 6 and 7, we use the Friedman test followed by the Nemenyi post-hoc test (Demšar, 2006). The streaming systems which perform within the confidence range of the top method are shown in bold.

In both TREE and RBF, the full AiRStream system using the repair component achieves the highest κ_{150} score. Both AiRStream variants as well as ARF score significantly higher than the DYNSE system in both datasets. As well as concept drift detection, DYNSE incorporates a dynamic classifier selection method which selects the most suitable ensemble based on the location of each observation in feature space. The synthetic datasets tested here display global drift where the entire feature space changes. This may explain the lower performance of this method. Both AiRStream variants perform better than ARF in both datasets. ARF is not able to identify or exploit recurring concepts once the model is dropped from the ensemble. The lower score it receives indicates this lack of long term memory harms its performance. In both the TREE and RBF datasets the full version of AiRStream including the repair component achieved a higher κ_{150} score than the variant with no repair functionality. This indicates that the repair component was able to improve the performance of the system. In order to show that this improvement is due to selecting more suitable models, we look at the C-F1 measure.

Table 3 Effectiveness of inference (κ_{150}) and condition inference (C-F1) on Synthetic data

System	Radial basis function		Random tree	
	κ_{150}	C-F1	κ_{150}	C-F1
NC	14.26 (11.98)	0.48*	1.14 (4.81)	0.48*
OK	−1.75 (3.20)	0.48*	0.05 (0.76)	0.48*
RF†	10.16 (9.02)	0.48*	36.10 (14.93)	0.48*
ARF	**59.27 (14.08)**	0.48 (0.07)	**50.68 (8.65)**	0.46 (0.05)
DYNSE	36.68 (9.63)	0.47 (0.08)	38.30 (9.63)	0.51 (0.08)
AS_b	**59.44 (11.13)**	**0.71 (0.07)**	**53.27 (8.90)**	**0.63 (0.08)**
AS_r	**59.59 (11.22)**	**0.76 (0.09)**	**54.76 (8.82)**	**0.71 (0.07)**

Values in parenthesis show standard deviation. Values within the confidence interval of the best method in each column are bolded

*These systems are not adaptive, so C-F1 is constant given the same proportion of concepts

†RF is a batch method rather than a streaming method

Table 4 Effectiveness of AQI inference

System	Rangiora		Arrowtown		Beijing	
	MRR	κ statistic	MRR	κ statistic	MRR	κ statistic
NC	89.90 (5.09)	45.36 (10.79)	97.33 (1.84)	**24.81 (7.59)**	56.31 (2.33)	10.61 (3.22)
IDW	59.77 (7.63)	3.90 (7.23)	91.98 (1.78)	14.96 (11.06)	58.03 (8.65)	6.98 (13.95)
NORM	79.82 (12.17)	28.96 (21.08)	96.34 (1.40)	17.32 (11.41)	61.14 (18.14)	27.20 (26.08)
OK	57.59 (5.25)	15.94 (1.03)	96.35 (0.79)	10.32 (3.87)	65.01 (3.60)	23.58 (5.60)
RF†	88.91 (5.46)	45.02 (10.93)	97.33 (1.84)	**24.81 (7.58)**	56.31 (2.33)	10.61 (3.22)
ARF	89.06 (5.13)	44.75 (10.83)	**98.04 (1.38)**	16.57 (8.56)	61.68 (5.44)	15.39 (9.35)
DYNSE	85.02 (6.38)	32.39 (20.09)	97.62 (1.71)	5.64 (5.22)	54.53 (3.32)	6.55 (4.42)
AS_b	**90.73 (4.07)**	**59.95 (7.56)**	**97.91 (1.50)**	13.18 (7.63)	**82.37 (3.16)**	**57.03 (7.42)**
AS_r	**90.74 (4.05)**	**60.11 (7.65)**	**97.94 (1.46)**	12.92 (7.58)	**82.42 (3.28)**	**57.11 (7.70)**

Values in parenthesis show standard deviation. Values within the confidence interval of the best method in each column are bolded

†RF is a batch method rather than a streaming method

6.4.2 Condition inference

The C-F1 measure in Table 3, shows the relationship between the model use in a system compared to ground truth context in the stream as described in Sect. 5. In these synthetic streams context is fully described by the function used to generate the ground truth label for each concept.

The NC, OK and RF baseline methods only use a single model across the stream so achieve a constant score. For ARF and DYNSE we take model use as the most influential member of the ensemble for each observation. In the RBF stream both ARF and DYNSE achieve a similar C-F1 score to the static methods. This indicates that in both methods the use of a certain model does not relate to the underlying data stream context. In the TREE

dataset, DYNSE achieves a higher score, indicating that it is able to form some relationship between model use and context.

In both streams, both AiRStream variants significantly outperform all baseline methods in terms of C-F1, indicating that model use has a much stronger relationship to the context of the data stream. We also see that AS_r outperforms AS_b, indicating that the repair component allows the system to build model histories more related to context dynamics. We show in Sect. 6.7 how this model history can be used for condition inference.

6.5 Real-world results

In this experiment we answer research question (2): how does the performance of AiRStream and its repair component translate to a real-world task? We use a real-world air quality inference task to compare the classification performance of AiRStream compared to baseline methods. The air quality inference highlights the need for systems capable of adapting to concept drift.

6.5.1 Classification performance

Table 4 displays the performance of each method in inferring $PM_{2.5}$ levels on the *Rangiora*, *Arrowtown* and *Beijing* real-world datasets. The first 20,000 observations are used for training. For streaming systems we report prequential performance measures on the remaining observations. The RF method does not train further, so report holdout performance measures. In this experiment we use the Mean Reciprocal Rank (MRR) and κ statistic performance measures. MRR reduces the penalty for classifications close to the true label, making it a better performance measure than accuracy for the ordered air quality levels used in these datasets. As shown in Fig. 4 all three datasets have imbalanced classes. The κ statistic is a measure of accuracy above a chance classifier (Žliobaité et al., 2015) which we report in order to minimize the effect of class imbalance. In the *Rangiora* and *Arrowtown* datasets, the No Change (NC) classifier achieved a strong performance in the MRR and κ statistic measures. This indicates that air pollution levels displayed a strong auto-correlation, i.e., the level of the target sensor was stable for large portions of the dataset. We also see that the standard inference methods IDW, NORM and OK show lower MRR and κ statistic in these datasets, indicating that the spatio-temporal relationships between sensors may be too complex for these simple methods to describe. In both datasets, we find that AiRStream performs well. In the *Rangiora* dataset, both AiRStream variants significantly outperform all baselines in both MRR and κ statistic. In the *Arrowtown* dataset, AiRStream is competitive with ARF and outperforms all other baselines using the MRR measure. As shown by the strong performance of NC, *Arrowtown* features very consistent levels for long periods of time. This makes it hard to produce results greater than baseline. In the *Beijing* dataset, both AiRStream variants significantly outperform all baselines for both MRR and κ statistic. The repair component increases both MRR and κ statistic for all datasets except for *Arrowtown*, where it showed an increase in the MRR measure. However, the benefit of repair is smaller than in synthetic datasets. This may be due to the short timescale of these datasets which restricts the amount of concept recurrence available. For example, both *Rangiora* and *Arrowtown* last less than three months which does not allow for seasonal recurrence. The adaptation repair mechanism helps increase the robustness of recurrence detection, so the lack of long term recurrence in the datasets may have

limited its benefit. To report performance on unbalanced classes, Table 5 shows a confusion matrix for *Rangiora* inferences which was averaged across all target sensors, for AS_r and RF. RF uses a similar tree based classification method to AiRStream, but as a batch method is trained on a smaller static training set. As an online method, AiRStream observes all training data, but only uses a portion of training data to build a model at each concept drift. On average, over the *Rangiora* dataset, each model used in AiRStream is trained on 3156 observations while RF is trained on a static set of 20,000 observations. Predictions for both methods are distributed across all $PM_{2.5}$ levels except level 5. AiRStream displays better performance at classifying $PM_{2.5}$ levels 0, 1, 3 and 4. This indicates that although each AiRStream model individually observes less training data than RF, the ability to adaptive select which observations to train on increases classification performance across a data stream. Neither make any level 5 predictions. The $PM_{2.5}$ distribution in Fig. 4 shows there were only 420 observations of levels 4 across all sensors in *Rangiora* over the evaluation period, which we believe was not enough training data to predict level 4 accurately. There were no observations of air quality level 5 over the evaluation period.

6.6 Runtime analysis

Tables 6 and 7 show the runtime of our method compared to the streaming baselines on synthetic and real data. DYNSE was run on an alternate framework, MOA, using a

Table 5 Average confusion matrix of $PM_{2.5}$ inferences for *Rangiora*

Levels	AiRStream					
	Predicted					
	0	1	2	3	4	5
0	**19652.92**	2041.52	105.916	34.05	1.23	0.00
1	1017.77	**5380.31**	589.55	91.71	1.11	0.00
2	665.08	610.66	**2372.41**	206.30	0.63	0.00
3	50.52	152.60	270.34	**529.34**	2.73	0.00
4	6.90	0.01	3.42	18.15	**3.83**	0.00
5	0.00	0.00	0.00	0.00	0.00	**0.00**
Levels	Random forest (RF)					
	Predicted					
	0	1	2	3	4	5
0	**19355.54**	2017.89	307.50	151.31	3.36	0.00
1	2039.86	**4418.30**	392.69	227.34	2.28	0.00
2	383.67	537.63	**2791.94**	141.44	0.41	0.00
3	204.61	296.79	163.90	**337.30**	2.94	0.00
4	0.61	7.15	3.30	19.72	**1.52**	0.00
5	0.00	0.00	0.00	0.00	0.00	**0.00**

Values in parenthesis show standard deviation. Values within the confidence interval of the best method in each column are bolded

Table 6 Runtime and memory on synthetic data

System	Radial basis function		Random tree	
	Runtime (s)	Memory (KB)	Runtime (s)	Memory (KB)
NC	**4.94 (0.16)**	**0.03 (0.00)**	**5.14 (0.15)**	**0.03 (0.00)**
OK	211.94 (39.18)	7.56 (3.08)	134.53 (2.16)	9.23 (0.00)
RF†	5.22 (0.18)	2029.95 (0.00)	6.96 (0.11)	2822.41 (2.03)
ARF	215.03 (16.96)	2166.31 (1274.04)	282.35 (13.57)	8967.27 (310.04)
AS_b	33.50 (1.64)	1315.90 (409.82)	44.41 (1.97)	3520.35 (704.83)
AS_r	72.97 (5.71)	1102.28 (372.49)	90.21 (7.25)	3215.85 (666.52)

Values in parenthesis show standard deviation. Values within the confidence interval of the best method in each column are bolded

†RF is a batch method rather than a streaming method

Table 7 Runtime of AQI inference

System	Rangiora Runtime (s)	Arrowtown Runtime (s)	Beijing Runtime (s)
NC	**23.66 (0.90)**	**30.01 (0.77)**	**9.98 (0.48)**
IDW	**26.96 (1.17)**	35.81 (0.92)	**11.64 (0.62)**
NORM	88.92 (4.83)	86.10 (3.67)	42.57 (2.35)
OK	591.36 (58.15)	233.32 (13.27)	387.09 (23.92)
RF†	**26.72 (1.05)**	39.65 (0.64)	11.74 (0.47)
ARF	410.29 (33.39)	496.36 (39.66)	364.03 (23.02)
AS_b	138.80 (9.60)	95.31 (5.64)	89.51 (5.01)
AS_r	156.39 (12.61)	111.17 (9.06)	99.36 (5.81)

Values in parenthesis show standard deviation. Values within the confidence interval of the best method in each column are bolded

†RF is a batch method rather than a streaming method

different language, Java. As this has an impact on the runtime and memory use of the system, DYNSE is not able to be compared to the other methods. While our proposed repair component incurs a small performance penalty over the base method, overall runtime is much smaller than the state-of-the-art method ARF. As an ensemble method, ARF runs all ensemble models, 10 in our experiments, at every time step. Our repair algorithm runs only K, five in our experiments, additional inactive models only during the testing period. Our method also uses substantially less memory than ARF, with the repair algorithm contributing only the size of one model, a_b, to the storage overhead.

6.7 Case study: air quality condition inference

In this section we answer research question (3): can the hidden underlying conditions of a data stream be inferred from the model use of a recurring concept system? We investigate how the models used by AiRStream can be used to infer environmental conditions in a real-world task. We note that in the real-world there can be many aspects of context

which change during concept drift. Different classification systems may react to different aspects. In this work we have access to only a few aspects for analysis. This means that an exhaustive comparison of frameworks is not possible, as the concept drift detected by some frameworks may not relate to the contextual aspects we analyse. In contrast to the quantitative comparison on synthetic data we present in Sect. 6.4.2, here we present a case study to highlight how certain known aspects of environmental conditions underlying the *Rangiora*, *Arrowtown* and *Beijing* datasets relate to model use.

Inferring air quality has been shown to be highly reliant on current weather conditions. We hypothesized that a data stream approach would allow changes in these conditions to be adapted to by a system without access to meteorological conditions. In this section we verify whether the changes detected by our system are related to changes in weather. We note that these weather conditions were not available to any system during training or testing, and are only used in post-hoc analysis.

6.7.1 Environmental data

The *Rangiora*, *Arrowtown* and *Beijing* datasets each present a different set of environmental conditions available to be monitored.

1. *Rangiora* We collected the current wind speed and direction from the weather station closest to each sensor for each observation.
2. *Arrowtown* We collected two time based features indicating if the observation was taken during daylight and if the observation was taken during the weekend. Here we take daylight hours as between 10 am and 5 pm, and the weekend as saturday and sunday.
3. *Beijing* We used the wind speed and direction, temperature and pressure from the weather station closest to each sensor for each observation.

Each source of environmental data was discretized into 8 equal density levels. Environmental level observations were linked to sensor readings by matching timestamps rounded to the nearest minute. We compare environmental conditions to the active model used by a system. For systems which use a single classifier to classify each observation, like AiRStream, we take the active model at a given observation as the classifier used to classify that observation. For ensemble systems, like DYNSE, we take the active model as the classifier which contributes most to the classification at each observation. In the case of a tie, we take the more common model to allow for more model reuse to be detected.

6.7.2 Measures of relationship

We use two measures to investigate the relationship between active model and environmental condition level. We first measure the strength of the relationship between each active model and each environmental condition level using the C-F1 score as discussed in Sect. 5. This is an F1 score between the active model used by AiRStream and a given environmental condition averaged across all discretized levels.

We also measure the ability to predict the level of a given environmental feature from observing the systems active model. We train a random forest classifier to classify the current level of each environmental feature using the current active model. We refer to the

accuracy of this system on a given environmental feature as ρ_m in Table 8. We compare this to the accuracy of a random forest classifier trained to predict the current level using current $PM_{2.5}$ readings, referred to as ρ_f, to find the additional performance in condition inference gained by using AiRStream. In both cases we compute accuracy using 10-fold cross validation. To maximise the performance of this baseline we use all sensors with no masking so ρ_f displays no variation across repetitions.

6.7.3 Condition inference results

Table 8 shows ρ_f, ρ_m and C-F1 for each environmental condition for the data sets they are available. We compare the performance of AiRStream active models to the active models found in the DYNSE system, which also captures recurring concepts. We highlight that this case study is limited to investigating only a few known aspects of the full context underlying each stream, and thus should not be taken as a performance comparison.

In the *Rangiora* dataset, Table 8 shows a P_f score of 34.87 for wind direction and 33.26 for wind speed. This indicates that a random forest model trained on sensor readings alone is able to achieve an accuracy of 34.87% predicting wind direction and 33.26% predicting wind speed. A random forest trained to predict environmental levels based on which model is selected as active by AiRStream is able to achieve accuracies of 50.39% and 49.41% respectively, shown by the P_m score under AS_r. This indicates that the active model selected by AiRStream is a better indicator of wind direction and speed in *Rangiora* than the raw sensor readings alone. The active model in DYNSE achieves a lower accuracy than using features, indicating that the active model here is not related to the underlying environmental conditions we investigate. We observe that AiRStream achieves C-F1 scores of approximately 0.47 for both conditions, while DYNSE achieves 0.24. This indicates some level of predictive ability from the models produced by our system. This adds validity to the hypothesis that our system can react to changes in environmental conditions without requiring additional input features.

We note that the AiRStream active model does not correspond particularly highly with temperature, pressure, wind direction and wind speed in *Beijing*. We hypothesize that this is due to a mismatch between the speed of change in these conditions and

Table 8 Condition inference on Real Data sets

Data set	Condition	P_f	AS_r		DYNSE	
			P_m	C-F1	P_m	C-F1
Rangiora	Wind direction	34.87*	50.39 (7.93)	0.47 (0.06)	33.41 (4.85)	0.24 (0.04)
Rangiora	Wind speed	33.26*	49.41 (7.97)	0.47 (0.06)	32.02 (4.34)	0.24 (0.04)
Arrowtown	IsDaytime	74.01*	74.46 (3.72)	0.35 (0.09)	58.20 (6.51)	0.14 (0.01)
Arrowtown	IsWeekend	71.12*	67.81 (6.51)	0.39 (0.08)	59.41 (6.20)	0.20 (0.02)
Beijing	Wind direction	14.52*	14.70 (0.40)	0.21 (0.02)	13.99 (0.19)	0.13 (0.01)
Beijing	Wind speed	35.92*	22.29 (2.89)	0.31 (0.04)	16.54 (1.25)	0.17 (0.02)
Beijing	Temperature	34.03*	27.02 (4.14)	0.33 (0.03)	30.70 (4.51)	0.23 (0.02)
Beijing	Pressure	32.68*	26.30 (3.97)	0.33 (0.03)	25.10 (3.97)	0.23 (0.02)

Values in parenthesis show standard deviation

*ρ_f displays no variation across repetitions

observation frequency, which was 1 hour in *Beijing* compared to 1 minute in other data sets.

Further results for the variant of AiRStream without repair, AS_b, are shown in "Appendix C". We see that both AiRStream variants are able to build a relationship between model use and wind in *Rangiora* greater than that of DYNSE, with repair achieving a slightly stronger relationship.

6.7.4 Multivariate condition visualization

Changes in environmental conditions come from complex interactions between many factors, so it is unlikely the models constructed by our system will perfectly match with any one feature. For example, rather than a single model relating to northerly wind, our system may find a model relating to northerly wind at high wind speed and another relating to northerly wind at low wind speeds. The previous evaluation does not consider the relationship between the model used and *sets* of environmental conditions. Figure 9 visualises the relationship between the active model and the combination of wind speed and direction. We plot each observation in the *Rangiora* data set as the vector with direction given by wind direction and magnitude given by wind speed. We color each data point by the active model chosen by the system (referred to by an unique ID). Certain combinations of wind speed and direction can be seen to be related to certain active models. For example strong south westerly wind is largely classified by Model ID 0 in Fig. 9.
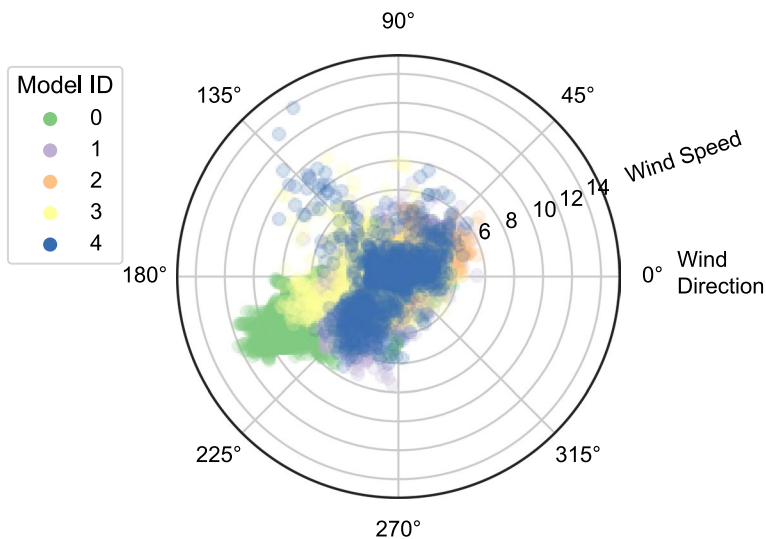

Fig. 9 Weather conditions compared to system model. Angle of each observation is given by wind direction, magnitude from (0, 0) is given by wind speed and color is given by the active model (referred to by a unique ID)

7 Related work

This section discusses adaptive approaches to concept drift proposed in the literature. A discussion on non-streaming approaches to air quality inference follows.

Streaming methods of adapting to changes in the distribution of data can be roughly divided into two groups, *blind* and *informed* (Gama et al., 2014). Blind approaches adapt to recent observations without explicitly detecting concept drift. A common example is using a sliding window of recent observations to train a model (Widmer and Kubat 1996). The most recent observations captured in the window are assumed to describe the distribution new observations will be drawn from, while older observations are forgotten. Another example is the Concept-Adapting Very Fast Decision Tree (CVFDT) (Hulten et al., 2001), which incrementally builds a decision tree using the Hoeffding bound to determine when to grow a new branch. Alternative branches are grown from recent data and replace older branches which are forgotten. Controlling how quickly observations are forgotten is a trade-off between stability and adaptability. Retaining observations for longer leads to a more stable classifier, which is important in stationary conditions, but can be slow to react to change.

Informed approaches explicitly detect changes in distribution. Explicit detection allows for a faster reaction to concept drift by removing old observations quickly while also retaining more relevant data when no concept drift is detected (Klinkenberg & Renz, 1998). Under the Probably Approximately Correct (PAC) framework, during stationary conditions the error rate of a classifier will decrease as the number of observations seen increases (Valiant, 1984). A change in the distribution of data can then be detected as a significant increase in the error rate of a classifier. Many drift detection methods, such as the Drift Detection Method (DDM) (Gama et al., 2004), Early Drift Detection Method (EDDM) (Baena-García et al., 2006) and Adaptive Windowing (ADWIN) (Bifet & Gavalda, 2007) are based on this idea. A common informed approach incorporating drift detection is to incrementally build a classifier while no detection has occurred, i.e. the distribution of data is stationary. When concept drift is detected, the current classifier is forgotten and a new classifier is built on incoming data representing the new distribution (Gama et al., 2004). Forgetting concepts is inefficient if the concept reoccurs in the future and must be relearned. This is important in many real-world cases where underlying contexts often reoccur, e.g. seasonal weather conditions or business cycles.

Recurring concept methods assume that concepts may reoccur over the data stream and previously learned relationships may become relevant again. A common characteristic of these methods is to store a set of previously used classifiers (Gonçalves Jr & De Barros, 2013; Alippi et al., 2013; Gama & Kosina, 2014). If a concept reoccurs, for example if the wind shifts to a previously seen direction, a stored classifier may be used to achieve higher performance than rebuilding a new classifier. Classifier reuse also provides an indicator that context, or the conditions underlying a data stream, are similar to a previous point in the stream (Gomes et al. 2010). Context information can be mined for further data, for example linking concept drift to environmental conditions (Borchani et al., 2015). Examples utilizing recurring concepts include the Recurrent Concept Drift (RCD) framework (Gonçalves Jr & De Barros, 2013), which stores a set of previously used classifiers in a repository. When a concept drift is detected, a statistical test compares the distribution of a recent window of observations to a sample of observations associated with each stored classifier. A similar distribution indicates that the stored classifier is relevant to the concept emerging in incoming data and should be reused. The Just In Time (JIT) framework

(Alippi et al., 2013) tests the performance of each stored classifier on a recent window to determine suitable candidates for reuse, while Gama and Kosina (2014) train a meta-classifier to predict the most suitable classifier for each observation.

The dynamic selection area of research also attempts to select the most suitable classifier for each observation (Cruz et al., 2018). While concept drift adaption attempts to select a classifier based on the underlying hidden context an observation is drawn from, dynamic selection attempts to select a classifier based on the location in feature space an observation is drawn from. Some methods, such as the Dynamic Selection Based Drift Handler (DYNSE) (De Almeida et al., 2016) framework combine both approaches, selecting a classifier both similar in context and feature space to the current observation. In both cases, the method of evaluating which stored classifiers are suitable for the current observation is important. Dynamic selection literature proposes several methods, including accuracy on a local region of samples, distance weighted accuracy, and K Nearest Oracles (KNORA) (Ko et al., 2008).

In order to conduct our condition inference analysis, in this work we make the assumption that context is discrete and concept drift is global, or covers the entire feature space. Under these assumptions, only the context of an observation rather than its location in feature space is relevant to classifier selection. In both recurrent concept drift adaption and dynamic selection classification can be conducted using either a single classifier or an ensemble (Gama et al., 2014; Cruz et al., 2018). A third assumption for our condition inference analysis is that model use is discrete. This assumption means that we select a single classifier for each observation rather than use an ensemble.

The adaptive learning problem that we consider in this paper could be considered to be a subset of lifelong learning (Parisi et al., 2019) problems. In lifelong learning, a system should dynamically adapt to new tasks while transferring learning from previously seen tasks. In adaptive learning, each distinct concept could be considered similar to a task in lifelong learning, with some key differences. Firstly, concept boundaries and properties are unknown in our problem, requiring concept drift detection and model selection. In some lifelong learning problems it is known when a new task occurs and whether the task is novel or a recurrence. Secondly, we consider the feature and label spaces to be shared between all concepts. This is similar to *homogeneous* lifelong learning (Weiss et al., 2016). Finally, in this work we do not consider transfer between concepts. In many lifelong learning problems, the classification performance of the system on observations drawn from one task is expected to increase as other tasks are learned. Here we consider concepts to be discrete, and avoid both positive and negative transfer between concepts.

Recent non-streaming work on air quality prediction have focused on urban locations, incorporating the many sources of environmental features available in these areas. We instead investigate rural locations where these features are not available. Zheng et al. (2013, 2015) separated spatial, temporal and meteorological features, feeding subsets of features into a spatial ANN model, a temporal linear regression or conditional random field model, and an inflection prediction model. The outputs of these models are merged based on current meteorological features. Meteorological features used include category of weather (sunny, overcast, etc.), humidity, wind speed and wind direction, as well as the weather forecast at a particular location. The authors noted that the use of these conditions is especially important in detecting inflection points where prediction patterns change. The link between inflection points and changes in weather, is however, not investigated. Yi et al. (2018) also utilized meteorological features, integrating them with spatial and temporal features using a deep fusion network. They noted that these 'indirect' features affect spatial and temporal transmission patterns, however, they do not attempt to detect such

changes explicitly. Cheng et al. (2018) incorporated meteorological features, weather, temperature, pressure, humidity and wind as well as points of interest and road network features into an attention based neural network model. They investigated feature importance, but did not compare feature importance under differing weather conditions. The attention mechanism is similar to our drift adaption method, whereby we attempt to find the most relevant features for each point in time. A critical difference is our method performs this in a low-information environment. Hsieh et al. (2015) investigated an offline method of constructing an 'AQI Affinity graph' measuring the relationship between sensor readings over time. They used additional traffic and point of interest features, for example the location of factories, to find similar sensors when constructing this graph. Change in spatial or temporal relationships is also not considered. Shang et al. (2014) investigated inferring traffic pollution from road network features, while Devarakonda et al. (2013) monitored traffic pollution using mobile sensors. These approaches are not applicable in the rural environments studied here. Low traffic volume in these environments means wood burning is a more significant source of air pollution than vehicles.

Similar to Zheng et al. (2015), we attempt to detect and adapt to inflection points, however we do this without knowledge of the relevant weather conditions. This allows our system to be used in more locations and without relying on hand picked weather features. This also allows us to adapt to drift in features not previously investigated, such as changes in sensor sensitivity.

Other methods attempted condition inference with only spatial and/or temporal readings. Hu et al. (2016) developed a 3D $PM_{2.5}$ interpolation system. A random walk approach models pollution transmission between 3D grid cells, with transmission rates learned from data. No change detection was implemented, and their results showed large performance variation as weather changed. Hu et al. (2019) used a Gaussian interpolation to infer $PM_{2.5}$ readings without the use of environmental features. Similar to previous research, no method of detecting or adapting to change were investigated. Li et al. (2017) investigated a method of mining causality patterns between sensors to determine propagation patterns and locate sources. The timing of uptrend events is matched across sensors to determine causality. In this research, no considerations were made to capture changes in these causality patterns over time. They concluded that the propagation patterns they found had no relationship to meteorological conditions.

8 Discussion and future work

In this work we hypothesize that adaptation errors may occur during concept drift detection and model selection. Our results show that detecting and repairing adaptation errors leads to higher classification accuracy and stronger correlation between the adaptions made by a system and ground truth concept drift. These results validate our initial hypothesis, indicating that adaptation errors do occur in adaptive streaming systems, and that repairing these errors is able to improve performance.

We also hypothesize that the concept drift detections and adaptations made by a system are related to changes and recurrences in underlying environmental conditions. Our synthetic experiments showed an average F1 score of above 0.7 from relating model reuse in AiRStream to ground truth environmental context. In real-world data, we were able to predict wind speed and direction substantially above baselines using only model reuse. In both cases, adaptation error repair improved these scores. These

results validate our initial hypothesis, indicating that the adaptations made by a system are linked to changes in underlying context. Further, the improved performance when repair is added indicates that the robustness of the adaptation process contributes to the strength of this relationship.

These lessons learned point towards areas of future research. The performance gain when repairing adaptation errors indicates that improving the robustness of the adaptation process can lead to improvements in classification accuracy and ability to model changes in context. Future work improving this robustness may lead to further gains. Our results indicating that system adaptations to concept drift are related to changes in context indicates that these adaptations may offer further insight into stream behaviour. Past work has used meta-information describing stream behaviour to improve system performance, for example the volatility of underlying contexts has been used to tune drift detection sensitivity (Huang et al., 2015). Investigating how the meta-information describing a systems adaptations to concept drift can be integrated into other aspects of the system may be valuable future work.

9 Conclusion

This paper proposes an algorithm to detect and repair errors in the concept drift adaption process. By sampling the performance of inactive models, we can identify cases where the active model is unsuitable for the current stream. Importantly, we can reset training done over this period. This increases the performance of our AiRStream system, as well as allowing the construction of a robust model of the concept drifts across a stream for further analysis. A real-world study classifying air quality levels in two rural towns shows that AiRStream outperforms eight baselines. Our evaluation shows indicates that the process of adapting to concept drift is related to the dynamics of the environmental conditions underlying the stream. Analysis of meteorological conditions in one of the locations shows that the concept drift model produced by AiRStream can be used to predict wind speed and direction with above 48% accuracy, 14.8% higher than using air quality readings alone. We believe that this is an important result for further investigation. The concept drift model produced by AiRStream has the potential to allow the inference of environmental conditions in locations where they were previously unavailable. Relating environmental conditions to concept drift also identifies environmental factors not currently included in the model which would be valuable to incorporate. For future work we plan to leverage the changes detected by AiRStream to further improve inference performance, for example, by automatically tuning parameters based on learned transition patterns. We also plan to investigate the generality of AiRStream by applying it to other tasks affected by change in unknown features. This could involve investigation of alternate or domain specific confidence measures for drift repair.

Appendix A: Significance testing

Figure 10 shows the critical difference diagrams obtained from performing the Nemenyi posthoc significance test for Tables 3 and 4.
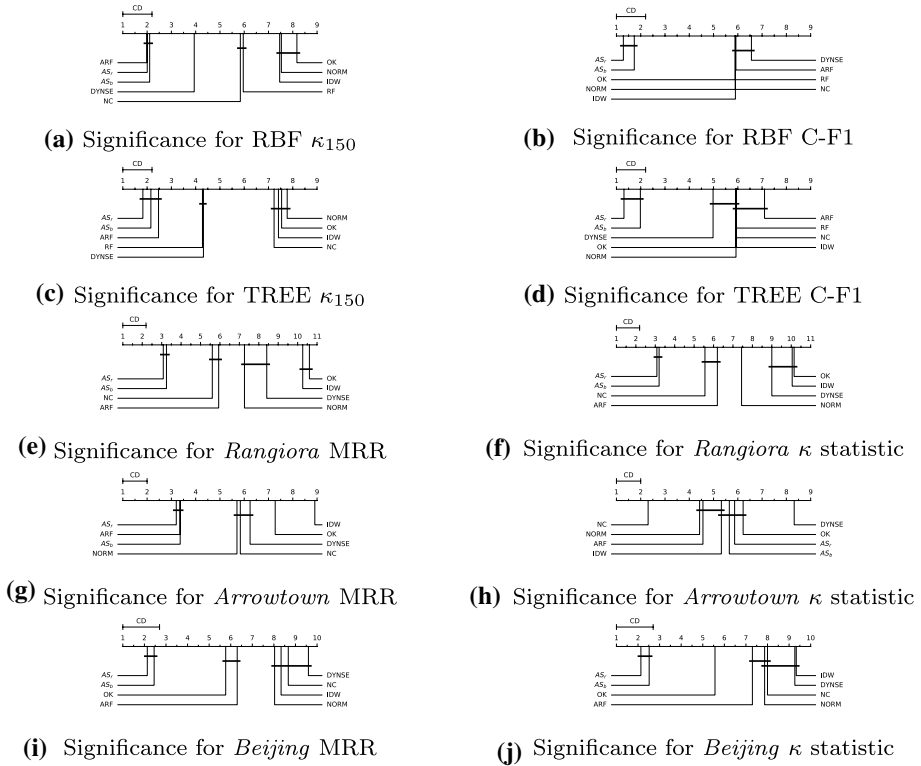
(a) Significance for RBF κ_{150}

(b) Significance for RBF C-F1

(c) Significance for TREE κ_{150}

(d) Significance for TREE C-F1

(e) Significance for *Rangiora* MRR

(f) Significance for *Rangiora* κ statistic

(g) Significance for *Arrowtown* MRR

(h) Significance for *Arrowtown* κ statistic

(i) Significance for *Beijing* MRR

(j) Significance for *Beijing* κ statistic

Fig. 10 Critical difference diagrams for Nemenyi significance test results

Appendix B: Distribution of full datasets

Figure 4 displays the distribution of air quality levels across the sensors selected for each dataset, over the observations used for evaluation. In our evaluation we select a subset of 9 target sensors in Rangiora due to breakage, and the initial 20,000 observations of each dataset are held back to train batch methods. Figure 11 shows the
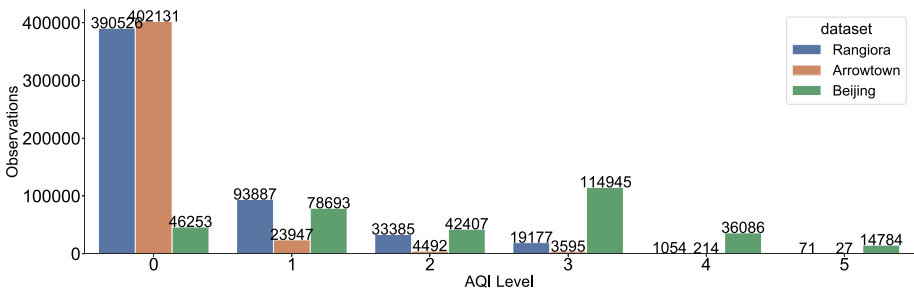

Fig. 11 Distribution of full datasets

Table 9 Condition inference on real data sets

Data set	Condition	P_f	AS_r		AS_b	
			P_m	C-F1	P_m	C-F1
Rangiora	Wind direction	34.87*	50.39 (7.93)	0.47 (0.06)	50.01 (9.60)	0.47 (0.06)
Rangiora	Wind speed	33.26*	49.41 (7.97)	0.47 (0.06)	40.05 (8.52)	0.47 (0.05)
Arrowtown	IsDaytime	74.01*	74.46 (3.72)	0.35 (0.09)	72.79 (4.36)	0.38 (0.10)
Arrowtown	IsWeekend	71.12*	67.81 (6.51)	0.39 (0.08)	68.98 (5.75)	0.41 (0.09)
Beijing	Wind direction	14.52*	14.70 (0.40)	0.21 (0.02)	14.85 (0.33)	0.22 (0.02)
Beijing	Wind speed	35.92*	22.29 (2.89)	0.31 (0.04)	23.20 (2.92)	0.33 (0.04)
Beijing	Temperature	34.03*	27.02 (4.14)	0.33 (0.03)	26.18 (3.33)	0.35 (0.03)
Beijing	Pressure	32.68*	26.30 (3.97)	0.33 (0.03)	25.98 (3.65)	0.34 (0.03)

Values in parenthesis show standard deviation

*ρ_f displays no variation across repetitions

distribution of the full datasets. We note that no air quality level 5 labels are observed in *Rangiora* in evaluation, while 71 were found overall.

Appendix C: Case study full results

Table 9 shows the results of the variant of AiRStream without repair, AS_b, to the version with repair, AS_r. We note that conclusions regarding performance should not be drawn from this comparison as not all environmental conditions contributing to concept drift can be analysed. However, we observe an increase in ability to predict wind speed and direction in *Rangiora* when repair is included.

References

Alippi, C., Boracchi, G., & Roveri, M. (2013). Just-in-time classifiers for recurrent concepts. *IEEE Transactions on Neural Networks and Learning Systems*, *24*(4), 620–634.

Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., & Morales-Bueno, R. (2006). Early drift detection method. *Fourth International Workshop on Knowledge Discovery from Data Streams*, *6*, 77–86.

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

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: Massive online analysis. *Journal of Machine Learning Research, 11*, 1601–1604. http://portal.acm.org/citation.cfm?id=1859903.

Borchani, H., Martínez, A. M., Masegosa, A. R., Langseth, H., Nielsen, T. D., Salmerón, A., Fernández, A., Madsen, A. L., & Sáez, R. (2015). Modeling concept drift: A probabilistic graphical model based approach. In *International symposium on intelligent data analysis* (pp 72–83). Springer.

Cheng, W., Shen, Y., Zhu, Y., & Huang, L. (2018). A neural attention model for urban air quality inference: Learning the weights of monitoring stations. In *Thirty-second AAAI conference on artificial intelligence*.

Chiu, C. W., & Minku, L. L. (2018). Diversity-based pool of models for dealing with recurring concepts. In *2018 International joint conference on neural networks (IJCNN)* (pp. 1–8). IEEE.

Cruz, R. M., Sabourin, R., & Cavalcanti, G. D. (2018). Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, *41*, 195–216.

De Almeida, P. R. L., Oliveira, L. S., Britto, A. D. S., & Sabourin, R. (2016). Handling concept drifts using dynamic selection of classifiers. In *2016 IEEE 28th international conference on tools with artificial intelligence (ICTAI)* (pp. 989–995). IEEE.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Devarakonda, S., Sevusu, P., Liu, H., Liu, R., Iftode, L., & Nath, B. (2013). Real-time air quality monitoring through mobile sensing in metropolitan areas. In *Proceedings of the 2nd ACM SIGKDD international workshop on urban computing* (p. 15). ACM.

Frias-Blanco, I., del Campo-Ávila, J., Ramos-Jimenez, G., Morales-Bueno, R., Ortiz-Diaz, A., & Caballero-Mota, Y. (2014). Online and non-parametric drift detection methods based on Hoeffding's bounds. *IEEE Transactions on Knowledge and Data Engineering*, *27*(3), 810–823.

Gama, J., & Kosina, P. (2014). Recurrent concepts in data streams classification. *Knowledge and Information Systems*, *40*(3), 489–507.

Gama, J., Rocha, R., & Medas, P. (2003). Accurate decision trees for mining high-speed data streams. In *Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 523–528).

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In *Brazilian symposium on artificial intelligence* (pp. 286–295). Springer.

Gama, J., Žliobaité, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, *46*(4), 1–37.

Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., et al. (2017). Adaptive random forests for evolving data stream classification. *Machine Learning*, *106*(9–10), 1469–1495.

Gomes, J. B., Menasalvas, E., & Sousa, P. A. (2010). Tracking recurrent concepts using context. In *International conference on rough sets and current trends in computing* (pp. 168–177). Springer.

Gonçalves, P. M, Jr., & De Barros, R. S. M. (2013). RCD: A recurring concept drift framework. *Pattern Recognition Letters*, *34*(9), 1018–1025.

Hsieh, H. P., Lin, S. D., & Zheng, Y. (2015). Inferring air quality for station location recommendation based on urban big data. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 437–446). ACM.

Hu, Y., Dai, G., Fan, J., Wu, Y., & Zhang, H. (2016). Blueaer: A fine-grained urban PM2. 5 3D monitoring system using mobile sensing. In *IEEE INFOCOM 2016-the 35th annual IEEE international conference on computer communications* (pp. 1–9). IEEE.

Hu, Z., Bai, Z., Bian, K., Wang, T., & Song, L. (2019). Implementation and optimization of real-time fine-grained air quality sensing networks in smart city. In *IEEE international conference on communications (ICC)* (pp. 1–6). IEEE.

Huang, D. T. J., Koh, Y. S., Dobbie, G., & Bifet, A. (2015). Drift detection using stream volatility. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 417–432). Springer.

Huggard, H., Koh, Y. S., Riddle, P., & Olivares, G. (2018). Predicting air quality from low-cost sensor measurements. In *Australasian conference on data mining* (pp. 94–106). Springer.

Hulten, G., Spencer, L., & Domingos, P. (2001). Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 97–106).

Klinkenberg, R., & Renz, I. (1998). Adaptive information filtering: Learning in the presence of concept drifts. *Learning for Text Categorization*, 33–40.

Ko, A. H., Sabourin, R., & Britto, A. S, Jr. (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, *41*(5), 1718–1731.

Li, X., Cheng, Y., Cong, G., & Chen, L. (2017). Discovering pollution sources and propagation patterns in urban area. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1863–1872). ACM.

Losing, V., Wersing, H., & Hammer, B. (2018). Enhancing very fast decision trees with local split-time predictions. In *2018 IEEE international conference on data mining (ICDM)* (pp. 287–296). IEEE.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, *113*, 54–71.

Shaker, A., & Hüllermeier, E. (2015). Recovery analysis for adaptive learning from non-stationary data streams: Experimental design and case study. *Neurocomputing*, *150*, 250–264.

Shang, J., Zheng, Y., Tong, W., Chang, E., & Yu, Y. (2014). Inferring gas consumption and pollution emission of vehicles throughout a city. In *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1027–1036). ACM.

United States Environmental Protection Agency. (2012). *Revised air quality standards for particle pollution and updates to the air quality index (AQI)*. United States Environmental Protection Agency: Tech. rep.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, *27*(11), 1134–1142. https://doi.org/10.1145/1968.1972.

Wackernagel, H. (2013). *Multivariate geostatistics: an introduction with applications*. Springer.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big data*, *3*(1), 1–40.

Widmer, G., & Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, *23*(1), 69–101.

Wong, D. W., Yuan, L., & Perlin, S. A. (2004). Comparison of spatial interpolation methods for the estimation of air quality data. *Journal of Exposure Science & Environmental Epidemiology*, *14*(5), 404–415.

Yi, X., Zhang, J., Wang, Z., Li, T., & Zheng, Y. (2018). Deep distributed fusion network for air quality prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 965–973). ACM.

Zhang, S., Guo, B., Dong, A., He, J., Xu, Z., & Chen, S. X. (2017). Cautionary tales on air-quality improvement in Beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *473*(2205), 20170457.

Zheng, Y., Liu, F., & Hsieh, H. P. (2013). U-air: When urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1436–1444). ACM.

Zheng, Y., Yi, X., Li, M., Li, R., Shan, Z., Chang, E., & Li, T. (2015). Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 2267–2276). ACM.

Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., & Holmes, G. (2015). Evaluation methods and decision theory for classification of streaming data with temporal dependence. *Machine Learning*, *98*(3), 455–482.

Authors and Affiliations

Ben Halstead[1] · Yun Sing Koh[1] · Patricia Riddle[1] · Russel Pears[2] · Mykola Pechenizkiy[3] · Albert Bifet[4,5] · Gustavo Olivares[6] · Guy Coulson[6]

Yun Sing Koh
ykoh@cs.auckland.ac.nz

Patricia Riddle
pat@cs.auckland.ac.nz

Russel Pears
russel.pears@aut.ac.nz

Mykola Pechenizkiy
m.pechenizkiy@tue.nl

Albert Bifet
abifet@waikato.ac.nz

Gustavo Olivares
Gustavo.Olivares@niwa.co.nz

Guy Coulson
Guy.Coulson@niwa.co.nz

1   School of Computer Science, The University of Auckland, Auckland, New Zealand

2   Auckland University of Technology, Auckland, New Zealand

3   Eindhoven University of Technology, Eindhoven, The Netherlands

4   University of Waikato, Hamilton, New Zealand

5   LTCI, Télécom Paris, IP-Paris, Paris, France

6   National Institute of Water and Atmospheric Research, Auckland, New Zealand