Reproducibility: Online Transfer Learning for Concept Drifting Data Streams

Helen McKay

1 INTRODUCTION

To ensure reproducibility, all BOTL and BOTL-C variants are available in an online repository[1]. In addition to the BOTL implementations, the drifting hyperplane and smart home heating synthetic data generators, and a sample of the real-world following distance data have been made available. In the following sections we provide more detail on the data generators and discuss the following distance data collection process.

2 EXPERIMENTAL SET-UP: DRIFTING HYPERPLANE DATA GENERATOR

For this benchmark data generator, an instance at time t, x_t, is a vector, $x_t = \{x_{t_1}, x_{t_2}, \ldots, x_{t_n}\}$, containing n randomly generated, uniformly distributed, variables, $x_{t_n} \in [0, 1]$. For each instance, x_t, a response variable, $y_t \in [0, 1]$, is created using the function $y_t = (x_{t_p} + x_{t_q} + x_{t_r})/3$, where p, q, and r reference three of the n variables of instance x_t. This function represents the underlying concept, c_a to be learnt and predicted. Concept drifts are introduced by modifying which features are used to create y. For example, an alternative concept, c_b, may be represented by function $y_t = (x_{t_u} + x_{t_v} + x_{t_w})/3$, where $\{p, q, r\} \neq \{u, v, w\}$ such that $c_a \neq c_b$. We introduce uniform noise, ±0.1, by modifying y_t for each instance x_t with probability 0.1.

A variety of drift types have been synthesised in this generator including sudden drift, gradual drift and recurring drifts. A sudden drift from concept c_a to concept c_b is encountered immediately between time steps t and $(t+1)$ by changing the underlying function used to create y_t and $y_{(t+1)}$. A gradual drift from concept c_a to c_b occurs between time steps t and $(t + m)$, where m instances of data are observed during the drift. Instances of data created between t and $(t+m)$ use one of the underlying concept functions to determine their response variable. The probability of an instance belonging to concept c_a decreases proportionally to the number of instances seen after time t while the probability of it belonging to c_b increases as we approach $(t+m)$. Recurring drifts are created by introducing a concept c_c that reuses the underlying function defined by a previous concept, c_a, such that we achieve conceptual equivalence, $c_c = c_a$.

Table 1 shows the performance of RePro, GOTL and BOTL variants on the drifting hyperplane data streams. RePro parameters $W_{max} = 30$, $\lambda_l = 0.05$, and $\lambda_d = 0.6$ were used across all frameworks. This meant that 30 instances of data were used to build models, instances were removed from the start of the sliding window if predictions were made within ±0.05 of the response variable, and drifts were detected when the R^2 performance of a model on the sliding window dropped below 0.6.

[1] https://github.com/XXXX

Table 1: Drifting Hyperplanes: predictions using RePro, GOTL, BOTL and BOTL-C variants for five sudden and five gradual drifting domains, where * indicates $p < 0.01$ in comparison to RePro and GOTL, and bold indicates the highest R^2 performance.

	Sudden Drift		Gradual Drift	
	R^2	RMSE	R^2	RMSE
RePro	0.950 (±0.001)	0.058	0.855 (±0.001)	0.070
GOTL	0.928 (±0.001)	0.069	0.825 (±0.001)	0.076
BOTL	***0.958** (±0.001)	0.053	***0.893** (±0.001)	0.060
BOTL-C.I	*0.957 (±0.001)	0.054	*0.886 (±0.003)	0.062
BOTL-C.II	*0.956 (±0.001)	0.054	*0.889 (±0.004)	0.061

2.1 Reproducibility

This data generator was designed to manipulate datasets such that BOTL could be evaluated on different types of concept drifts by changing various parameters. The parameters used to create datasets for this paper are displayed in Table 2 with a brief description of how each parameter can be used to manipulate the data. We created each instance, x_t, using random numbers between 0 and 1 to represent each feature. As discussed previously, all concepts were created using the function $y_t = (x_{t_k} + x_{t_l} + x_{t_m})/3$, however, for future work, this can be altered or used alongside other functions, such as cosine. The number of features used by the concept function can also be modified using this data generator, however all results presented in this paper use three features to calculate the ground truth. Drift type, drift length, concept transitions and concept length are all parameterised, enabling BOTL to be evaluated on data streams with sudden, gradual and recurring drifts of different durations and concept orderings. Low levels of uniform noise have been added with a small probability, however, this can be increased or decreased as necessary.

Additionally, the data generator has been designed to aid future work into domain adaptation for online transfer learning by including parameters to allow composite concepts, conflicting concepts and alternative concept functions to be used. A composite concept, c_{comp}, is created by combining multiple independent concepts by, for example, taking the mean, where

$$c_{comp} = \frac{1}{N} \sum_{i=1}^{N} c_i,$$

and a concept, c_{con}, that conflicts with concept c_a is created using

$$c_{con} = 1 - c_a.$$

Table 2: Drifting hyperplane data generator parameters.

Parameter	Description	Value
Features	Number of features	10
Drift type	Sudden or gradual concept drift	Both
Dependencies	Number of features concepts are dependent on	3
Concept functions	Number of functions used to create concepts	1
Noise probability	Probability of the target value being distorted by noise (±0.1)	0.1
Concepts	Number of independent concepts to be used	5
Concept length	Number of consecutive instances for each concept	500
Drift length	Length of gradual drifts	100
Concept schedule	Order of concept transitions allowing for recurrences	Repeating
Randomise order	Flag to randomise the order of concept transitions defined by the concept schedule	False
Shared concepts	The number of concepts shared with other data streams	≤ 3
Conflicting concepts	If true some concepts use 1 − (original function value)	False
Composite concepts	If true some concepts are created by combining values from multiple concepts	False

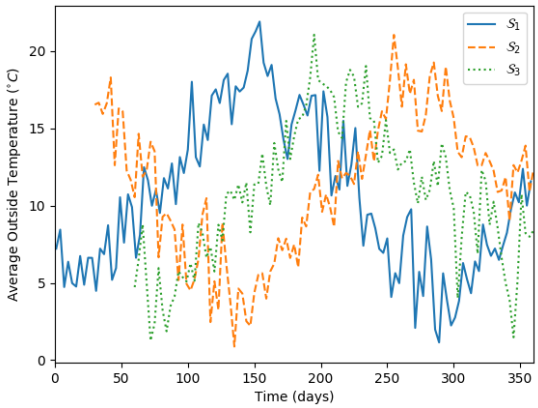


Figure 1: Average daily temperatures ($^{\circ}C$) used to induce concept drift within three heating simulation data streams.

Table 3: Heating Simulations: predicting desired heating temperatures across five domains using RePro, GOTL, BOTL and BOTL-C variants, where * indicates $p < 0.01$ in comparison to RePro and GOTL, and bold indicates the highest R^2 performance.

	R^2	RMSE
RePro	0.607 (±0.003)	2.601 (±0.015)
GOTL	0.709 (±0.003)	2.231 (±0.009)
BOTL	***0.786** (±0.009)	1.914 (±0.042)
BOTL-C.I	*0.779 (±0.010)	1.946 (±0.044)
BOTL-C.II	*0.744 (±0.008)	2.102 (±0.036)

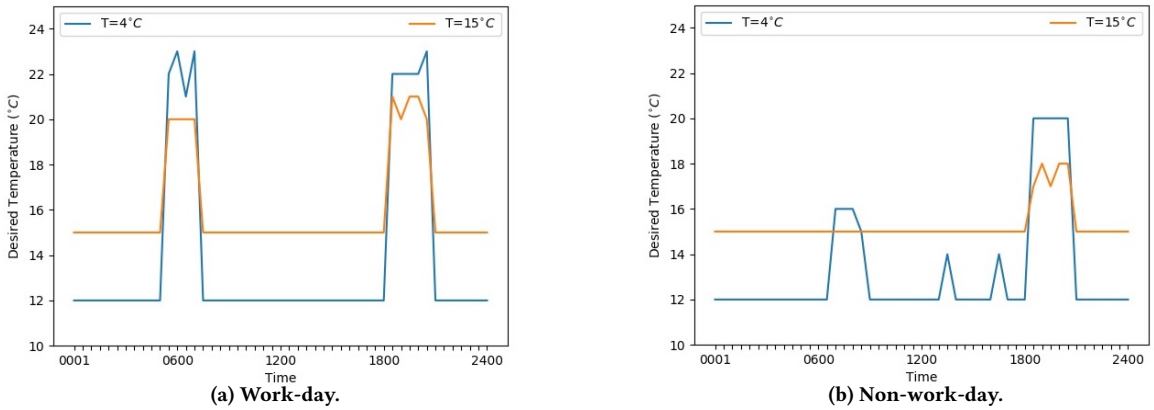
3 EXPERIMENTAL SET-UP: SMART HOME HEATING SIMULATION

A simulation of a smart home heating system was created, deriving the desired room temperature of a user. Heating temperatures were obtained from weather data collected from a weather station in Birmingham, UK, from 2014 to 2016. This dataset contained rainfall, temperature and sunrise patterns, which were combined with a schedule, obtained from sampling an individual's pattern of life, to determine when the heating system should be engaged. The schedule was synthesised to vary desired temperatures based on time of day, day of week, and external weather conditions, creating complex concepts. To create multiple domains, weather data was sampled from overlapping time periods and used as input to the synthesised schedule to determine the desired heating temperatures. Due to the dependencies on weather data, each stream was subject to large amounts of noise. Concept drifts were introduced manually by changing the schedule, however, drifts also occurred naturally due to changing weather conditions. Figure 1 displays the average daily outside temperature for three data streams. Each stream commences at different time intervals. Due to sampling weather data from overlapping time periods, and the seasonality of weather, data streams follow a similar pattern, ensuring predictive performance can benefit from knowledge transfer. By using complex concepts, dependent on noisy data, the evaluation of BOTL is more indicative of what is achievable in real-world environments.

Table 3 shows the performance of RePro, GOTL and BOTL variants on these data streams. We chose RePro parameters $\lambda_l = 0.5$, $\lambda_d = 0.6$, and $W_{max} = 700$, creating a sliding window that encapsulated approximately two weeks of heating and weather data, to analyse frameworks on this data. Instances were removed from the start of the sliding window if predictions were made within ±0.5° of the desired heating temperature, and drifts were detected when the R^2 performance of the target model on the current window of data dropped below 0.6.

3.1 Reproducibility

The smart home heating simulation is dependant on real-world weather data, collected from a UK weather station, and a synthetic heating schedule. Using external weather conditions such as temperature and rainfall, the schedule determines the desired household temperature for an individual. The schedule is divided into time periods with varying heating requirements, a simplified example is shown in Algorithm 1. The division of time periods is determined by the day of the week, which indicates work-days and non-work-days. Finally the schedule uses a minimum temperature which indicates the minimum temperature desired by the user regardless of external weather conditions. This temperature is recorded for all time periods that have not explicitly been defined by the schedule, except for occasions when the outside temperature is greater than this, in which case the outside temperature is recorded. Figure 2



(a) Work-day. (b) Non-work-day.

Figure 2: Example of desired household temperature for a work-day and a non-work-day assuming constant outside temperatures of 4°C and 15°C.

Algorithm 1 Work-day heating schedule.

```
Input: time, outside_temp, rain, min_temp
if 0500 < time < 0730 then
    return getMorningTemp(outside_temp,rain,min_temp)
else if 1800 < time < 2100 then
    return getEveningTemp(outside_temp,rain,min_temp)
else if 2100 < time < 2330 then
    return getLateTemp(outside_temp,rain,min_temp)
else
    return getAwayTemp(outside_temp,rain,min_temp)
```

illustrates an example of desired household temperatures for work-days and non-work-days over a period of 24 hours for two outside temperatures, 4°C and 15°C, assuming a constant temperature and no rainfall throughout. Creating a schedule in this way enables complex concepts to be created, capturing desired household temperatures that vary depending on external weather conditions and an individual's presence within the home.

Five data streams were created using overlapping periods of weather data and a schedule. The weather data and schedule used to produce the results presented in this paper are available in the online repository. The addition of weather data from other geographical regions and alternative heating schedules can be easily added to create a diverse set of domains, and will be used in future work investigating BOTL with domain adaptation.

4 EXPERIMENTAL SET-UP: FOLLOWING DISTANCE SAMPLE DATA

This dataset uses a vehicle's following distance and speed to calculate TTC when following a vehicle. Vehicle telemetry data such as speed, gear position, brake pressure, throttle position and indicator status, alongside sensory data that infer external conditions, such as temperature, headlight status, and windscreen wiper status, have been recorded at a sample rate of 1Hz. Additionally, some signals such as vehicle speed, brake pressure and throttle position were averaged over a window of 5 seconds to capture a recent history of vehicle state. Vehicle telemetry and environmental data can be used to predict TTC and used to personalise vehicle functionalities such as ACC by identifying the preferred following distance, reflecting current driving conditions. Data was collected from 4 drivers for 17 journeys which varied in duration, collection time and route. Each journey is considered to be an independent domain and BOTL enables knowledge to be learnt and transferred across journeys and between drivers. Each data stream is subject to concept drifts that occur naturally due to changes in the surrounding environment such as road types and traffic conditions.

Table 4: Following Distances: predicting TTC across seven domains using RePro, GOTL, BOTL and BOTL-C variants, where * indicates $p < 0.01$ in comparison to RePro and GOTL, and bold indicates the highest R^2 performance.

	R^2	RMSE
RePro	0.497 (±0.002)	0.636 (±0.002)
GOTL	0.619 (±0.001)	0.554 (±0.003)
BOTL	0.665 (±0.002)	0.524(±0.002)
BOTL-C.I	0.666 (±0.002)	0.523(±0.002)
BOTL-C.II	***0.673** (±0.001)	0.518(±0.002)

Table 4 shows the performance of RePro, GOTL and BOTL variants across seven data streams. The RePro parameters $\lambda_l = 0.1$, $\lambda_d = 0.5$ and $W_{max} = 80$ were used, encapsulating 80 seconds of vehicle data. Instances were removed from the start of the sliding window if predictions were made within ±0.1 seconds of the recorded TTC value, and drifts were detected when the R^2 performance of the target model on the current window of data dropped below 0.6.

4.1 Reproducibility

The following distance data streams were collected by recording vehicle telemetry and sensory data when driven by four different drivers in the UK. Two types of journeys, scripted and unscripted, were undertaken during this data collection process. Scripted journeys are comprised of mostly motorway and main road driving, whereas unscripted journeys recorded vehicle data for general purpose driving such as commuting. Using these two types of journeys to record data enabled a diverse set of concepts to be recorded

Table 5: Following Distances: predicting TTC on six publicly available domains using RePro, GOTL, BOTL and BOTL-C variants, where bold indicates the highest R^2 performance.

	R^2	RMSE
RePro	0.414 (±0.004)	0.671 (±0.004)
GOTL	0.331 (±0.010)	0.675 (±0.004)
BOTL	0.423 (±0.001)	0.667(±0.001)
BOTL-C.I	0.421 (±0.001)	0.668(±0.001)
BOTL-C.II	**0.449** (±0.001)	0.653(±0.001)

since driving style can be impacted by the purpose of a journey, familiarity of the route, external weather conditions and traffic conditions. Two drivers participated in scripted journey data collection, obtaining a total of 6 data streams, which have been made available online[2]. These data streams have been collected from two routes at varying times of day enabling the effects of route familiarity, weather conditions and traffic conditions to be captured within these six data streams alone. The remainder of the data streams used in this paper have been collected during general purpose driving, therefore capture a wider variety of concept drifts, however, for data protection reasons, this data cannot be made publicly available.

For reproducibility, we present Table 5, containing the results of using RePro, GOTL, BOTL and BOTL-C variants using only the publicly available data. We use the same parameter sets as discussed in Section **??**. The use of bi-directional transfer leads to an increased performance for these datasets, however, the performance increase is less significant, $p < 0.2$ and $p < 0.08$ for RePro and GOTL respectively, in comparison to the results presented in Table 4. This can be attributed to a reduction in diversity among the datasets. The driving styles of only two drivers on two routes are captured within these domains, highlighting the importance of transferring knowledge between a diverse set of domains so the target learner can be enhanced through the use of knowledge learnt from various driving styles in different environments.

[2]https://github.com/XXXX