OXFORD

Briefings in Bioinformatics, 2023, **24(1)**, 1–12
https://doi.org/10.1093/bib/bbac506
Problem Solving Protocol

Concept drift detection in toxicology datasets using discriminative subgraph-based drift detector

Vandana Bharti, Shabari S. Nair, Akshat Jain, Kaushal Kumar Shukla and Bhaskar Biswas

Corresponding author. Kaushal Kumar Shukla, Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi 221005, Uttar Pradesh, India. Tel: +91-5427165313; E-mail: kkshukla.cse@iitbhu.ac.in

Abstract

Due to the increasing importance of graphs and graph streams in data representation in today's era, concept drift detection in graph streaming scenarios is more important than ever. Contributions to concept drift detection in graph streams are minimal and practically non-existent in the field of toxicology. This paper applied the discriminative subgraph-based drift detector (DSDD) to graph streams generated from real-world toxicology datasets. We used four toxicology datasets, each of which yielded two graph streams – one with abrupt drift points and one with gradual drift points. We used DSDD both with the standard minimum description length (MDL) heuristic and after replacing MDL with a much simpler heuristic SIZE (number of vertices + number of edges), and applied it to all generated graph streams containing abrupt drift points and gradual drift points for varying window sizes. Following that, we compared and analyzed the results. Finally, we applied a long short-term memory based graph stream classification model to all the generated streams and compared the difference in the performances obtained with and without detecting drift using DSDD. We believe that the results and analysis presented in this paper will provide insight into the task of concept drift detection in the toxicology domain and aid in the application of DSDD in a variety of scenarios.

Keywords: concept drift, toxicology, Graph streams, graph stream classification, DSDD

Introduction

Graphical representation of information and graph mining is finding increasing relevance in today's world due to the emergence of several complex networks like social acquaintance networks, organizational networks, metabolic networks, etc. Consequently, learning and classification algorithms on graphs and graph streams [1–5] have piqued the interest of researchers, and several contributions have been made to this field. However, most of these algorithms assume a stationary setting [1–3]. They are designed for environments where the informational content or distribution of data does not change. This is not the case in the practical world. The phenomenon of changing data distribution due to some "hidden context" in machine learning is called concept drift and is a very real occurrence. For example, social acquaintance networks are quite susceptible to distribution change as the interactions of each individual can change due to arbitrary reasons, resulting in new relations, connections, and, consequently, a changed network. Thus, concept drift can result in degradation of the model performance as the model trained on historical data is just not relevant for prediction on the new data. Hence it is imperative that concept drift be detected and dealt with accordingly.

Concept drift detection approaches can be broadly classified into two categories – active approaches and passive approaches. Active approaches are based on the methodology of detecting the drift first and then triggering some kind of adaption mechanism according to the detected drift. On the other hand, passive approach models get continuously updated irrespective of the drift detected. Aside from that, a passive approach can be based on a single model or ensemble model. Coming to traditional data streaming scenarios, there are several active and passive approaches available. Just in time classifier (JIT)

Vandana Bharti She is a Ph.D. research scholar and teaching assistant in the Department of Computer Science and Engineering at IIT-BHU Varanasi, India. She has authored and co-authored several peer-reviewed articles. Her research interests include nature-inspired optimization, multiobjective optimization, hyperspectral imaging, pattern classification, machine learning and deep learning.

Shabari S. Nair He is a B.Tech student in the Department of Computer Science and Engineering at IIT-BHU Varanasi, India. His research interests include pattern classification, machine learning and deep learning.

Akshat Jain He is a B.Tech student in the Department of Computer Science and Engineering at IIT-BHU Varanasi, India. His research interests include pattern classification, machine learning and deep learning.

Kaushal Kumar Shukla He is professor of Computer Science and Engineering at the IIT (BHU), Varanasi. He has 38 years of research and teaching experience. Prof. Shukla has published more than 160 research papers in reputed journals and conferences. 18 PhDs have been awarded under his supervision. He has been the Head of the CSE department for 6 years as well as Dean (Faculty Affairs) at the IIT (BHU). He holds 24 Intellectual Property Rights in the area of socially relevant computing. He has to his credit, many projects of national importance. Before joining IIT, he was with Hindustan Aeronautics Ltd and Smiths Aerospace UK. He was a member of the joint working group for DECU development of the Light Combat Aircraft Tejas. Prof Shukla is a Fellow of the Institution of Engineers, Fellow of the Institution of Electronics and Telecommunications Engineers, and Senior Member, ISTE.

Bhaskar Biswas He is an associate professor with Department of Computer Science and Engineering, Indian Institute of Technology (BHU). Dr. Biswas received the Ph.D. degree in computer science and engineering from Indian Institute of Technology (BHU), Varanasi, India. He has authored or co-authored more than 100 national and international journal publications, book chapters and conference papers. His research interests include data mining, text analysis, online social network analysis, and evolutionary computation.

Received: August 4, 2022. **Revised:** September 29, 2022. **Accepted:** October 24, 2022

[6] and drift detection method (DDM) [7] are two examples of active approaches. Both of these work on the windowing adaption mechanism. For the passive approaches, concept-adapting very fast decision tree (CVFDT) [8] is a single model-based approach, whereas the Streaming Ensemble Algorithm [9] is an ensemble-model based passive approach.

In the graph streaming scenario, there is a scarcity of literature pertaining to algorithms capable of detecting concept drift, where drift is related to the changing informational content or pattern of the graphs. In fact, to the best of our knowledge, there are only three notable approaches: graph entropy method (GEM) [10], discriminative subgraph-based drift detector (DSDD) [11] and Zambon *et al*. [12]. GEM is a supervised approach in which we calculate the entropy for subgraphs in a window with respect to their class membership. It then uses cumulative sum on entropy sequence to determine significant shift. Zambon *et al*. uses a kind of mapping to convert graph streams to a numeric vector and then it applies a multivariate change detection test, consequently detecting concept drift. It is parametrically constrained by a prototype graph for conversion to a numeric vector. DSDD uses a sliding window based mechanism to find drift points. Contrary to GEM, DSDD is an unsupervised approach, and also, it is not bound to any type of constraint as *Zambon et al*. is. Moreover, DSDD has been found to provide better results in comparison to GEM and Zambon *et al*.

Predicting the toxicity of chemical compounds has since long been a popular task in machine learning [13, 14]. The Tox21 Data Challenge 2014, in particular, aimed to crowdsource many such independent research works and analyze how well they can predict compounds that interfere in biochemical pathways. In this challenge, approximately 12 000 chemicals/drugs were used as data, and 12 different toxic effects were considered. Several machine learning-based methods were used to predict toxicity, which can be broadly classified into similarity-based methods such as nearest neighbor approaches and feature-based methods such as linear models, random forest models and naive bayes models. Deep learning methods like Deeptox [13] have also been presented. To the best of our knowledge, concept drift in toxicology datasets remains an unexplored topic. It is natural for predictive models trained on toxicology datasets to experience concept drift.

In this paper, we have used the DSDD [11] on graph streams generated from toxicology datasets. These graph streams are simulated so as to contain abrupt as well as gradual concept drift points. We have applied DSDD with the standard minimum description length (MDL) [15] heuristic as well as after replacing it with a rather simpler heuristic – SIZE, on all the generated graph streams containing abrupt drift points and gradual drift points for varying window sizes, and have presented a comparative study as well as analyzed the obtained statistics. Following that, we applied a long short-term memory (LSTM)-based graph stream classification model to the generated streams and compared the performance obtained without detecting drift to those obtained after adapting the model to the detected drift (using DSDD).

Motivation and contribution: The primary role of concept drift detection is to make any model adaptable to changing distributions and relationships between inputs and outputs. If a model trained on old data is used on new data after it has undergone concept drift, its performance may suffer significantly. The predictive model trained on old data is simply insufficient and irrelevant for making predictions on new data. As a result, it is crucial that concept drift be detected in both streaming and non-streaming scenarios, and the model is made adaptive to such drifts.

We believe that concept drift detection can be extremely useful in the field of toxicology. As previously discussed, predictive analysis on toxicology datasets is an important task with numerous applications. Consequently, drift detection in this field is also quite relevant. The importance of drift detection in the toxicology domain can be better understood by considering a scenario as follows: Consider that we have a predictive model for predicting whether a particular compound is active or inactive and that this model was trained on an old sample of compounds. Now suppose that, for unknown reasons that were not previously considered, the entire coming sample has a different distribution, e.g. some inherent relationship between certain factors has changed in such a way that resulted in a drastically higher number of active compounds, or maybe in such a way that a class of, previously mostly inactive compounds, have started becoming regularly active. Such changes could indicate a concept drift. In such cases, if the drift is not detected and the model is not adapted to this change, it could lead to poor results, which could have serious economic and safety implications. And, because the domain of toxicology is frequently intertwined with that of pharmacology, biology, biochemistry, etc., we believe that detecting concept drift in toxicology datasets is a very useful and relevant task.

The key motivations and contributions of this work can be summarized as follows:

- Concept drift has appealed to us as a machine learning concept that could have adverse implications if not handled properly. We have been quite intrigued with detecting concept drift in graph streams primarily because of the dynamic nature of graphs and the way it can be used to represent a lot of data in today's world. Hence, we have taken the concept drift detection on graph streams as our primary topic.
- Among the drift detection algorithms for graph streams we have come across, DSDD has proven to have the best performance overall. So we have chosen DSDD as our drift detection method.
- We believe that drift detection in this particular domain could have very useful applications, as discussed before.
- The performance of DSDD in detecting gradual drift points in real-life graph streams was not examined in [11]. Thus we have applied DSDD for gradual drift detection in graph streams on real-world toxicology datasets.
- Only one method (MDL) for discriminative subgraph extraction was used in [11]. In order to see how MDL compares to a simpler heuristic, we used another heuristic (SIZE) and compared its performance to that of MDL.
- We have also presented comparative results with different window sizes 30, 50, 150, 300, and evaluated the entire process using three performance metrics, namely drift detection rate (DDR), delay of detection (DoD) and average rate of false alarm (FA1000).
- By comparing the graph stream classification performances obtained with and without using DSDD to detect drifts and adapt the predictive model accordingly, we have presented a clear insight into the impact and usefulness of drift detection in the toxicology domain.

The remainder of this paper is organized as follows: Section 2 (Preliminaries) will provide an overview of the preliminary concepts and terms closely related to this study. Section 3 (Methodology) details the workflow as well as describes the process of graph stream creation. The experimental findings are presented and analyzed in Section 4 (Experimental results). Finally, the paper is concluded in Section 5 (Conclusion).

Preliminaries

Before proceeding with the proposed work, we look at some of the preliminary terms, notations and methods.

Graph

A graph $G(V, E)$ is defined by two sets V and E, where $V = \{v_1, v_2, v_3, \ldots v_n\}$ is the set of n vertices of the graph and $E = \{d_1, d_2, d_3, \ldots d_m\}$ is the set of m edges of the graph. Each vertex v_i is denoted by $< NodeID >< NodeLabel >$ where *NodeID* is a unique identification assigned to each vertex, and *NodeLabel* is the label of the vertex which may or may not be unique. Each edge d_i is denoted by $< Type >< Source >< Dest >< EdgeLabel >$ where *Type* is a flag that indicates whether the edge is directed or undirected, *Source* and *Dest* indicate the *NodeIDs* of the source and destination vertices. *EdgeLabel* is the label of the edge, which may or may not be unique to each edge.

It is to be noted that here we are only considering attributed graphs where vertices and edges are associated with labels that are in the form of strings or numbers.

Graph stream

A graph stream G_s is defined as an infinite sequence of graphs $\{G_1, G_2, G_3, \ldots G_n \ldots\}$ that come at different times $\{t_1, t_2, t_3, \ldots t_n \ldots\}$ respectively. Here, a graph G_i is the graph that comes at time i.

Graph streams can be both online as well as offline. An offline graph stream is one whose all graphs and corresponding times of arrival are known beforehand, while for an online graph stream, the graphs come as the time gradually proceeds, i.e. we only know about the graphs that have come before the current time. We will be dealing with online graph streams for the rest of the paper.

Concept drift

Concept drift represents the change in behavior of a model to predict the target variable over time in an unforseen way.

In our case, we are concerned with concept drift in graph streams where the concept drift occurs due to the change in structure or pattern of graphs over time.

Long short-term memory

LSTM [16] is an RNN architecture that is capable of selectively remembering information for a long time. It solves the vanishing gradient problem of RNNs in which RNNs tend to forget earlier inputs of a sequence in the case of long sequences. This capability of LSTMs to learn long-term sequential information makes them well suited for the classification and prediction of time series data of possibly very long or unknown duration.

An LSTM maintains long-term memory at any particular time t using what is known as cell state c_t. That is, at time t, in addition to the hidden state h_{t-1} from the previous cell as well as the current input x_t, the previous cell state c_{t-1} is also taken by the LSTM unit. Three gates – forget gate, input gate and output gate, are used to regulate the information held in the cell state and calculate the hidden state. The forget gate c_t is used to decide what information to forget from the cell state. The input gate i_t determines what information should be remembered in the long-term memory, i.e. what information should be added to the cell state. The output gate o_t uses the cell state c_t, the input x_t as well as the previous hidden state h_{t-1} to calculate the current hidden state h_t. Equations (1–6) depict the various operations performed in one LSTM unit.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_t) \tag{1}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3}$$

$$c'_t = tanh(W_c[h_{t-1}, x_t] + b_c) \tag{4}$$

$$c_t = f_t * c_{t-1} + i_t * c'_t \tag{5}$$

$$h_t = o_t * tanh(c_t) \tag{6}$$

Discriminative subgraph-based drift detector

DSDD [11] is a concept drift detection approach on graph streams proposed by Ramesh Paudel and William Eberle. It is an unsupervised sliding window-based technique. It mainly consists of three steps.

1. **Discriminative subgraph generation:** At each time t we extract the best (say p) discriminative subgraphs for the graph G_t. Here, discriminative subgraph refers to that subgraph that best compresses the graph G_t using the MDL heuristic. It implies that the discriminative subgraph is the subgraph S that minimizes the following:

$$M(S, G) = L(G_t \mid S) + L(S) \tag{7}$$

   where G_t is the original graph, S is the subgraph, $L(G_t \mid S)$ is the description length of G_t after compressing it using S and $L(S)$ is the description length of the subgraph itself.

   We use the SUBDUE [17] algorithm to achieve the above. The implementation of SUBDUE used is available as part of the graph-based anomaly detection (GBAD) tool [18].
2. **Window entropy calculation:** DSDD considers the entropy or structural information content of a graph as a measure of its concept [19]. Here, we consider the entropy of a window as a measure of the distribution of the discriminative subgraphs of the graphs that are part of the window. We employ Shannon's formula [20] to calculate the entropy of each fixed-size window. Thus, we end up with a series of entropy values corresponding to each window, as the fixed-size window moves one step forward each time.
3. **Concept drift detection:** In DSDD, we use the entropy values of each window obtained to check whether concept drift has occurred or not. We use two consecutive retrospective subsequence samples obtained at time i and time $(i + n)$. We then find the Pearson diversion-based dissimilarity measure between these two consecutive samples using the direct density-ratio estimation approach called RuLSIF [21]. We call this measure to obtain the change score (CS). Finally, we compare the CS obtained with a user-defined threshold (δ).
   a. If $CS \geq \delta$, it means that drift is detected. This may be an actual drift point or just a false alarm.
   b. If $CS < \delta$, it means that drift is not detected.

Methodology

In this section, we first describe the source of the toxicology datasets we have used and briefly discuss their relevance to our specific task.

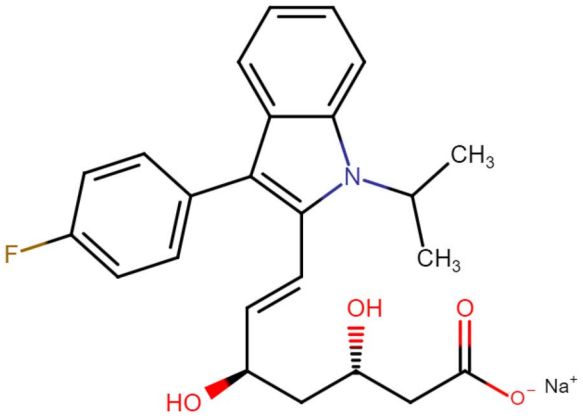

Figure 1. Fluvastatin sodium.

Datasets

We have used four datasets that have been provided as part of the training datasets of Tox21 Data Challenge 2014 [22]. The main aim of this challenge was to crowdsource the work of several researchers to observe how well they can predict the interference of some chemical compounds (using their structure only) in biochemical pathways. *The datasets are generated from nuclear receptor signaling, and stress pathway assays run against Tox21's 10 000-compound library (Tox21 10K) to build models and look for structure–activity relationships* [22]. In all of these datasets, each compound is represented as a graphical structure in which the elements like oxygen, nitrogen, fluorine, etc., are represented by nodes, with the node labels being the element symbols like O, N, F, etc. The various relations between the elements, like a single bond, double bond, etc., are represented by graphs edges, with the edge labels being –,=, etc. The graph labels indicate whether the compound is active (1) or inactive (0). We now briefly describe the four datasets.

1. **Tox21-Ahr:** Aryl hydrocarbon receptors (Ahrs) are a special class of receptors that are very crucial for adaptive responses to environmental changes, i.e. they make the cell more powerful against environmental changes, and they mediate cellular responses to environmental pollutants. There are some molecules that activate the Ahr signaling pathways. This dataset presents the various compounds and their structures and provides information on whether the compound activates Ahr signaling pathways or not. An example of an active compound in the Ahr dataset is shown in Figure 1.
2. **Tox21-ARE:** Oxidative stress (which is an imbalance between free radicals of oxygen and antioxidants in our body) is hazardous, and it contributes to dangerous diseases like cancer. The antioxidant response element (ARE) signaling pathways reduce this problem. This dataset presents the various compounds and their structures and provides information on whether the compound activates ARE signaling pathways or not. An example of an active compound in ARE dataset is shown in Figure 2.
3. **Tox21-MMP:** Mitochondrial membrane potential (MMP) is a parameter that describes the effectiveness of mitochondrial function inside the cell. It contributes to the generation of ATP and hence is important for various cellular processes. There are small molecules that disrupt the mitochondrial membrane potential, hence inhibiting mitochondrial functioning. This dataset is about molecules that disrupt the MMP. An example of an active compound in the MMP dataset is shown in Figure 3.

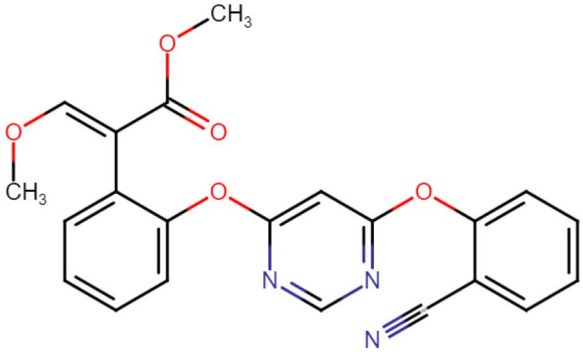

Figure 2. Azoxystrobin.

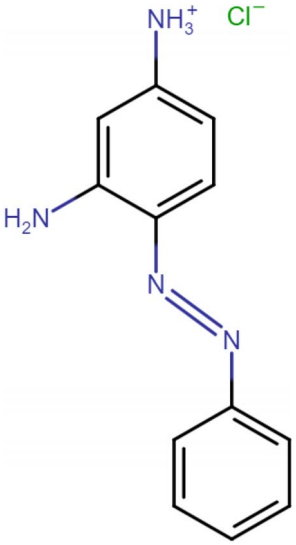

Figure 3. C.I. Basic orange 2.

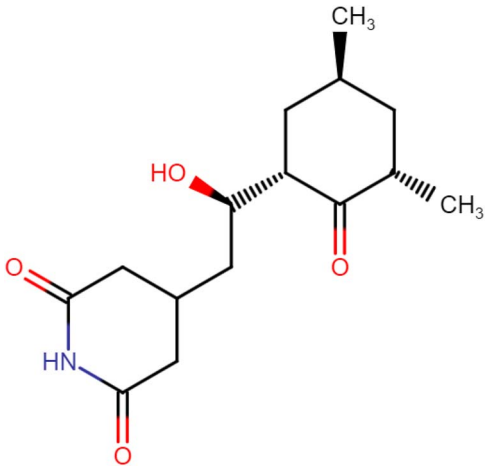

Figure 4. Cycloheximide.

4. **Tox21-Aromatase:** Our body contains two vital hormones – Androgen and Estrogen. Aromatase acts as a catalyst that converts androgen to estrogen and tries to maintain the balance of androgen and estrogen hormones in our body. This dataset provides various molecules and tells whether the molecules inhibit aromatase or not. An example of an active compound in this dataset is shown in Figure 4.

Table 1. Graph streams with abrupt drift

Graph stream	Total graphs	Drift points
Tox21-Ahr	2400	800,1600
Tox21-ARE	2120	1060
Tox21-MMP	2000	1000
Tox21-Aromatase	640	320

Table 2. Graph streams with gradual drift

Graph stream	Total graphs	Drift points
Tox21-Ahr	1558	600
Tox21-ARE	1084	200
Tox21-MMP	2010	900
Tox21-Aromatase	1000	360

Graph stream preparation

We have created graph streams for all four datasets, simulating two kinds of concept drifts in each dataset:

Abrupt concept drift: When the concept (data distribution or input–output relation) changes abruptly at a point in time, it is called abrupt concept drift.

Gradual concept drift: When the concept (data distribution or input–output relation) changes gradually and slowly over a period of time, it is called gradual concept drift.

We now look at the simulation procedure of abrupt and gradual drift points for the considered datasets.

Abrupt

1. **Tox21-Ahr** For this dataset, we create a graph stream consisting of a total of 2400 graphs with two drift points at times 800 and 1600. To simulate this, we send graphs of the two classes (active and inactive) in a 1:19 ratio up to 800. The ratio is then reversed, and graphs in a 19:1 ratio are sent up to 1600. Then we again reverse the ratio and send them in a ratio of 1:19 up to 2400. Clearly, there is an abrupt change in distribution at 800 and 1600, thus causing concept drifts at these points.
2. **Tox21-ARE** Here, we simulate a graph stream of 2120 graphs with a drift point at 1060. For this, we send graphs in a ratio of 19:1 (active: inactive) up to 1060 and reverse the ratio after that up to 2120.
3. **Tox21-MMP** Here, we simulate a graph stream of 2000 graphs. We maintain a ratio of 1:19 (active: inactive) up to 1000 and reverse it after that. This gives a concept drift at 1000.
4. **Tox21-Aromatase** Here, we simulate a rather small graph stream of 640 graphs with a relatively early drift point of 320. We do this in the same way as MMP, i.e. by maintaining a ratio of 19:1 (active: inactive) and reversing it after that.

The graph streams generated are summarized in Table 1.

Gradual

1. **Tox21-Ahr** Here, we create a graph stream with a total of 1558 graphs and a gradual drift point at 600. We do so by sending graphs in a ratio of 1:5 (active: inactive) up to 600, then in 1:4 from 600 to 630, then in 1:3 from 630 to 658, and in 1:2 after that. This results in a gradual concept drift which starts at 600 and stabilizes after 658.
2. **Tox21-ARE** Here, we simulate a graph stream of a total of 1084 graphs with a gradual drift at 200. We create this by maintaining a ratio of 1:1 (active:inactive) up to 200, 1:2 from 200 to 206, 1:3 from 206 to 214, 1:4 from 214 to 224, 1:5 from 224 to 236, 1:6 from 236 to 250, 1:7 from 250 to 266, 1:8 from 266 to 284 and finally 1:9 from 284 to 1084. This results in a gradual drift that begins at 200 and stabilizes after 284.
3. **Tox21-MMP** Here, we simulate a graph stream with a total of 2010 graphs with a gradual drift at 900. For this, we send graphs in a ratio of 1:9 (active: inactive) up to 900, in 2:8 from 900 to 930, in 3:7 from 930 to 960, in 4:6 from 960 to 990, in 5:5 from 990 to 1020, in 6:4 from 1020 to 1050, in 7:3 from 1050 to 1080, in 8:2 from 1080 to 1110 and finally in 9:1 after that. This results in a gradual concept drift starting at 900, which stabilizes after 1110.
4. **Tox21-Aromatase** Here, we simulate a graph stream of a total of 1000 graphs with a gradual drift at 360. We do so by sending graphs in a ratio of 1:9 (active: inactive) up to 360, in 2:8 from 360 to 390, in 3:7 from 390 to 420, in 4:6 from 420 to 450 and 5:5 after that. This gives us a gradual drift that starts at 360 and stabilizes after 450.

The graph streams generated are summarized in Table 2.

Procedure

DSDD has been applied on the above created graph streams under standard conditions as well as after changing a few parameters. Before proceeding to the actual procedure, a basic understanding of the SIZE heuristic is important. SIZE simply refers to the sum of the number of edges and vertices of the graph. SIZE can be used as a replacement to MDL for discriminative subgraph generation using eq. (7), but instead of the description length of a graph, the SIZE of the graph is used. DSDD using both MDL as well as SIZE has been executed in order to compare the performance of a simpler heuristic such as SIZE with that of a more complex one like MDL.

The entire procedure of work can be divided into two experiments as follows:

1. Apply DSDD on graph streams with abrupt drift and gradual drift using standard MDL heuristic as well as SIZE heuristic with varying window sizes – 30, 50, 150, 300. Subsequently, we compare and analyze the results obtained. The steps are shown in Figure 5.
2. Apply an LSTM-based graph stream classification model on the streams and compare the performance obtained with and without incorporating drift detection using DSDD.

Graph stream classification

The usefulness of drift detection is only evident when some classification models have been applied on the datasets. This is what we strive to do in the second experiment. Here a predictive model has been applied to the graph streams and the performance obtained with and without incorporating drift detection using DSDD has been compared so as to know the importance of drift detection and the effectiveness of DSDD in improving the model performance. The predictions are made in a supervised fashion using an LSTM-based approach.

Firstly use an unsupervised approach to learn graph embeddings of the graphs via ground truth distances between the graphs. This approach has been inspired by UGraphEmb [23]. Here, we use the Laplacian spectrum distance, which is an efficient measure of similarity or ground truth distance between two

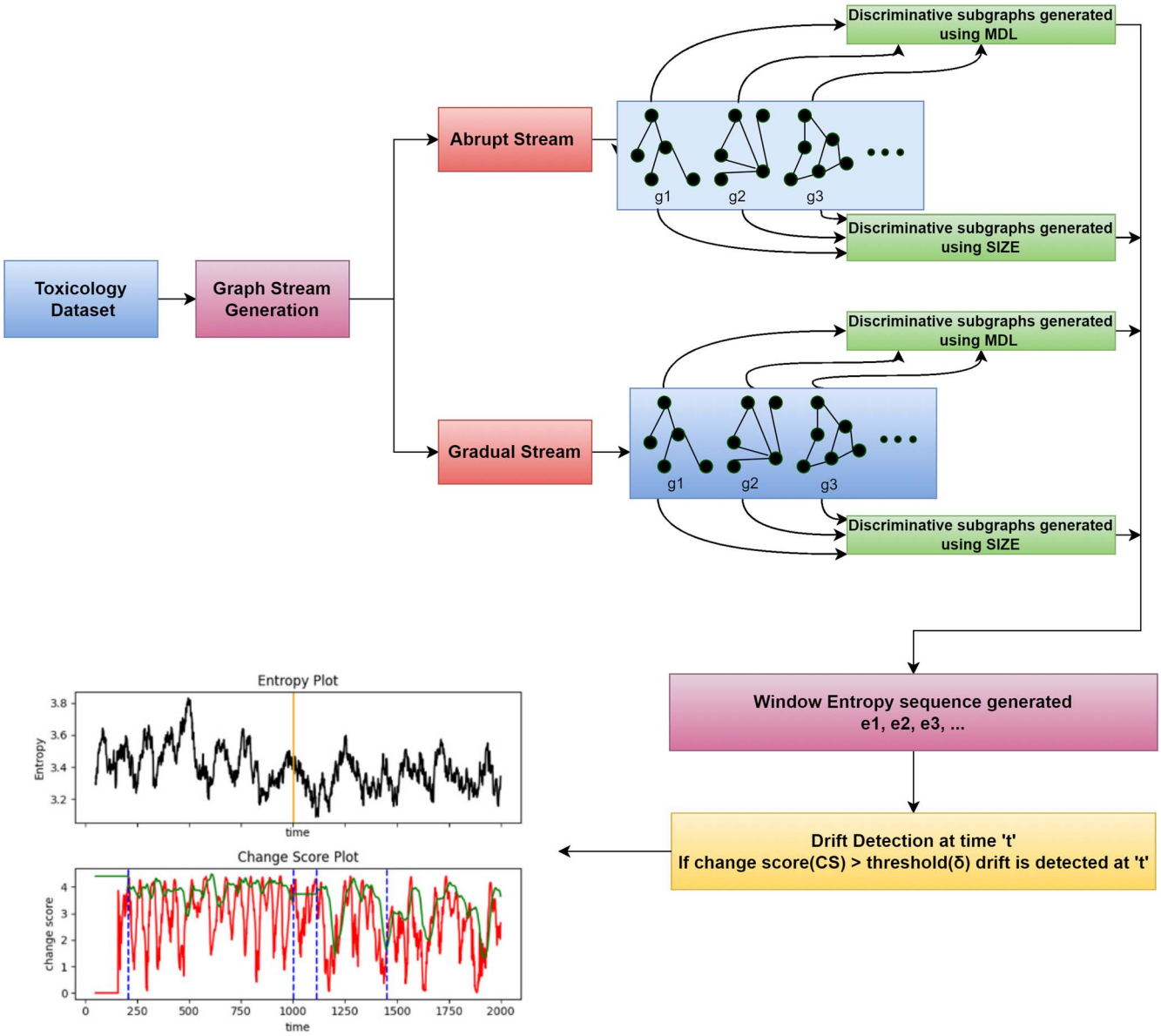

Figure 5. Diagrammatic representation for application of DSDD on graphs streams with abrupt and gradual drift using standard MDL heuristic as well as SIZE heuristic.

graphs. The embeddings of pairs of graphs are then simultaneously calculated so that the combination of the embeddings of each pair matches the ground truth distance between them.

Following this, we split the stream into training and testing sets. We use the embeddings of the first 80% of the graphs for training and the remaining 20% for testing. For the training set we fix a sentence width w, and create several overlapping sentences of graphs, of length w, like $\{g_1, g_2, g_3, ..., g_w\}$, $\{g_2, g_3, g_4, ..., g_{w+1}\}$, $\{g_3, g_4, g_5, ..., g_{w+2}\}$ and so on. Here we use overlapping sentences so as to increase the amount of training data as well as for better pattern learning from the sentences. We then sample 15% of these created sentences to form the validation set. Finally, an LSTM with a many-to-many architecture is trained on these sentences of embeddings to learn to predict the class (active or inactive) of each graph of a sentence. For testing, we form non-overlapping sentences of length w like $\{g_1, g_2, g_3, ..., g_w\}$, $\{g_{w+1}, g_{w+2}, g_{w+3}, ..., g_{2w}\}$ and so on. To accommodate for the possible lack of graphs in the last sentence, we use dummy graphs that have no meaning to fill the last sentence and bring its length to w. Since LSTM is unidirectional and since all the dummy graphs come after the actual graphs of the stream, the predictions made for the dummy graphs would in no way affect that of the actual graphs. We ignore the predictions made for the dummy graphs while calculating the performance. The proposed model is depicted in Figure 6.

Experimental results

Experimental setup

For the first experiment, DSDD was applied on each graph stream for a total of 50 iterations using the parameter values as suggested in [11]. The discovery of discriminative subgraphs was made through two heuristics – MDL and SIZE, both of which were done using the GBAD tool. All experiments were carried out on an Intel Xeon(R) Silver 4214 CPU @2.20GHz, 32GB RAM, and 16GB Quadro RTX 5000 PCIe accelerator, LINUX operating system.

For the second experiment, the same model parameters were taken for each stream to maintain fairness. The graph embeddings, which are vectors of 96 features, were calculated

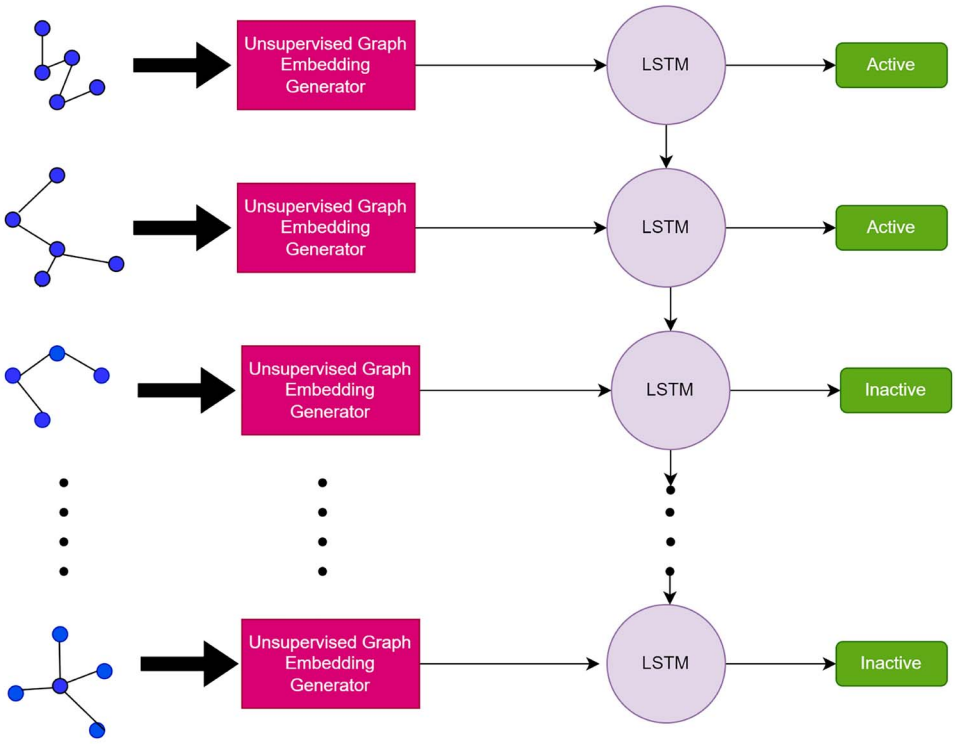

Figure 6. Proposed LSTM Model – A sentence of graphs is taken as input. Embeddings are generated for each graph of a sentence using an unsupervised approach. These embeddings are then given as input to an LSTM, and the class of each graph (active or inactive) is predicted.

using 500 randomly generated pairs of graphs from the stream. We have chosen a sentence length of 40 graphs for training and testing. The model has one hidden layer of 64 nodes, and the output layer uses the sigmoid activation function to make the prediction. We have used Binary Cross-Entropy loss and Adam optimizer for training, and the learning rate we have taken is 0.001. The models have been trained for 400 epochs with early stopping in order to avoid overfitting.

Performance metrics

Before proceeding to the actual results obtained, we briefly discuss the various metrics on which the performance of DSDD has been evaluated.

- **DDR:** This refers to the Drift Detection Rate. The higher the value of DDR, the better the performance. If x denotes the number of correctly detected drift points and t denotes the total number of drift points, then DDR is given by

$$DDR = \frac{x}{t} \tag{8}$$

- **DoD:** Delay of Detection refers to the average time that has passed before the algorithm has detected a true drift point. Lower this value, the better the performance. If D denotes the time at which a true drift point is detected by the algorithm, and A denotes the time at which the drift has actually occurred, then DoD is given by

$$DoD = Mean(D - A) \tag{9}$$

- **FA1000:** FA1000 refers to the average rate of false alarms raised every 1000 graphs. The lower the value of FA1000, the better the performance.

Results and discussion

As part of the first experiment, we first look at the DDR, DoD and FA1000 scores obtained by running 50 iterations for the four graph streams with abrupt drifts using the MDL heuristic as well as the SIZE heuristic for window sizes 30, 50, 150 and 300. The results are summarized in Table 3. The table shows the mean as well as the standard deviations obtained for each run. We have highlighted the best scores between MDL and SIZE pertaining to each metric for each window size and stream. It has been observed that the performance for each metric is generally better for smaller window sizes, with the best performance mostly held by the window size 30. This trend is observed for both SIZE and MDL. There are a few exceptions to this general trend, such as scores for the Aromatase stream for window size 300. It can be seen that the DoD score is 0 for SIZE in this case. Even the FA1000 scores are excellent for MDL as well as SIZE. But this cannot be taken into consideration as the DDR scores for both MDL and SIZE are quite close to 0. This means that hardly any drift has been detected. In such a case of virtually no drift detection, no matter how good the DoD and FA1000 scores are, they do not have any serious importance. Such low DDR scores could be due to the fact that the Aromatase stream has only 640 graphs with an abrupt drift at 320. These numbers are quite close to each other, as well as the window size of 300.

It should also be noted that even though there is a general increase in performance with decreasing window sizes, the execution time also increases with window size, as smaller window sizes require a larger number of iterations to cover the graph stream.

Further, we look at how the SIZE heuristic compares with the MDL heuristic. It has been observed that generally, the MDL heuristic is performing better and more consistently than the SIZE heuristic. But it is to be noted that SIZE is not too far behind MDL. In fact, there are several instances, such as window sizes 150 and

Table 3. Abrupt drift experiment results

Dataset	Window size	Heuristic	DDR	DOD	FA1000
Tox21-Ahr	30	MDL	**1.00 ± 0.00**	**11.43 ± 10.01**	**1.28 ± 0.10**
		SIZE	**1.00 ± 0.00**	13.70 ± 10.97	**1.28 ± 0.10**
	50	MDL	**1.00 ± 0.00**	**15.93 ± 10.83**	**1.42 ± 0.25**
		SIZE	**1.00 ± 0.00**	17.07 ± 11.33	**1.42 ± 0.25**
	150	MDL	0.89 ± 0.21	35.81 ± 24.69	**2.37 ± 0.49**
		SIZE	**0.95 ± 0.15**	**25.69 ± 19.43**	2.40 ± 0.52
	300	MDL	0.85 ± 0.23	32.61 ± 25.83	**2.02 ± 0.52**
		SIZE	**0.86 ± 0.24**	**24.55 ± 22.10**	2.02 ± 0.53
Tox21-ARE	30	MDL	**1.00 ± 0.00**	14.42 ± 14.46	**0.96 ± 0.09**
		SIZE	**1.00 ± 0.00**	**12.76 ± 17.33**	0.98 ± 0.13
	50	MDL	0.98 ± 0.14	**19.52 ± 18.88**	**1.03 ± 0.21**
		SIZE	**1.00 ± 0.00**	25.48 ± 23.78	1.10 ± 0.28
	150	MDL	0.94 ± 0.24	29.80 ± 33.25	2.24 ± 0.66
		SIZE	**0.96 ± 0.20**	**26.68 ± 25.35**	**2.21 ± 0.62**
	300	MDL	0.88 ± 0.32	29.52 ± 30.60	**1.96 ± 0.44**
		SIZE	**0.92 ± 0.27**	**26.30 ± 29.22**	2.04 ± 0.40
Tox21-MMP	30	MDL	**1.00 ± 0.00**	**17.04 ± 14.79**	1.04 ± 0.14
		SIZE	**1.00 ± 0.00**	19.32 ± 22.02	**1.02 ± 0.10**
	50	MDL	**1.00 ± 0.00**	**15.04 ± 19.19**	1.17 ± 0.26
		SIZE	**1.00 ± 0.00**	20.54 ± 20.98	**1.11 ± 0.23**
	150	MDL	**0.98 ± 0.14**	34.16 ± 29.47	2.23 ± 0.47
		SIZE	**0.98 ± 0.14**	**29.98 ± 30.28**	**2.13 ± 0.61**
	300	MDL	**0.94 ± 0.24**	**28.34 ± 31.07**	2.02 ± 0.51
		SIZE	0.92 ± 0.27	30.08 ± 26.64	**1.92 ± 0.39**
Tox21-Aromatase	30	MDL	**1.00 ± 0.00**	14.78 ± 15.70	**3.12 ± 0.00**
		SIZE	**1.00 ± 0.00**	**12.04 ± 16.24**	**3.12 ± 0.00**
	50	MDL	**1.00 ± 0.00**	**18.26 ± 20.65**	3.22 ± 0.37
		SIZE	**1.00 ± 0.00**	21.06 ± 22.81	**3.12 ± 0.00**
	150	MDL	0.54 ± 0.50	**50.14 ± 46.46**	1.62 ± 0.44
		SIZE	**0.62 ± 0.48**	52.36 ± 45.13	**1.53 ± 0.49**
	300	MDL	**0.02 ± 0.14**	2.12 ± 14.84	0.28 ± 0.60
		SIZE	0.00 ± 0.00	**0.00 ± 0.00**	**0.19 ± 0.51**

300 of Ahr stream, where SIZE surpasses MDL in performance. Also, SIZE is much simpler to calculate than MDL. Hence, SIZE can be considered as a computationally economic replacement to MDL in cases where we can tolerate some occasional decrease in performance and possibly some lack of consistency.

We have generated two graphs for each iteration for every window size and stream. The first is an entropy (vs) time graph. The second graph is a threshold and change score (vs) time graph. Two example graph sets are given in Figures 7 and 8. In the entropy (vs) time graph, the black line indicates the entropy, and the yellow vertical lines indicate the actual drift points. In the threshold and change score (vs) time graph, the red line indicates the change score, and the green line indicates the threshold. The vertical blue lines indicate all the drift points detected, be they actual drifts or false alarms. In Figures 7 and 8, we can see that at all points where drift has been detected, the change score is greater than the threshold.

We now look at DDR, DoD and FA1000 scores obtained by running 50 iterations for the four graph streams with gradual drifts using the MDL and the SIZE heuristic for window sizes 30, 50, 150, 300. The results are summarized in Table 4 and the best scores were highlighted in bold. The same conventions have been used as in the previous Table 3. Firstly it can be seen that, similar to abrupt drift streams, smaller window size generally leads to better performance.

Here we can also see that, in general, the scores of gradual drift are on par with their abrupt counterparts. Hence, it can be said that DSDD works quite well in the case of real-life toxicology

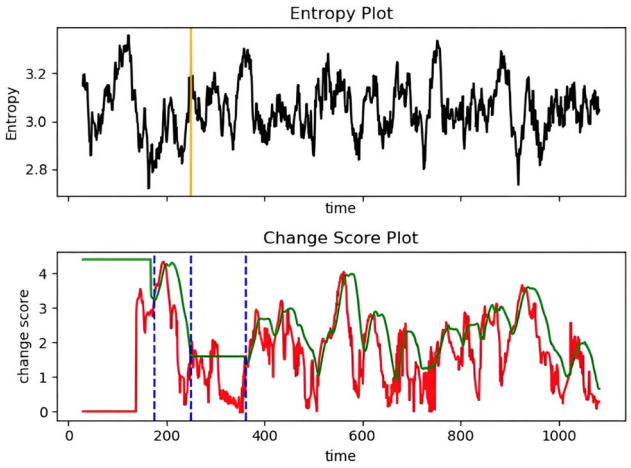

Figure 7. (Entropy vs time) and (change score vs time) plots for ARE abrupt stream using MDL heuristic, taking window size as 30.

gradual drift streams as well. However, it should be noted that, similar to the case of abrupt drift streams, there are some exceptions, like in the case of window size 300 for ARE stream. But these are most likely due to the window sizes being too close or even greater than the drift point and sometimes quite close to the stream size too.

It is also noted that MDL is generally performing better than SIZE. Hence it can be said that most of the trends observed for abrupt drift streams apply to gradual as well.

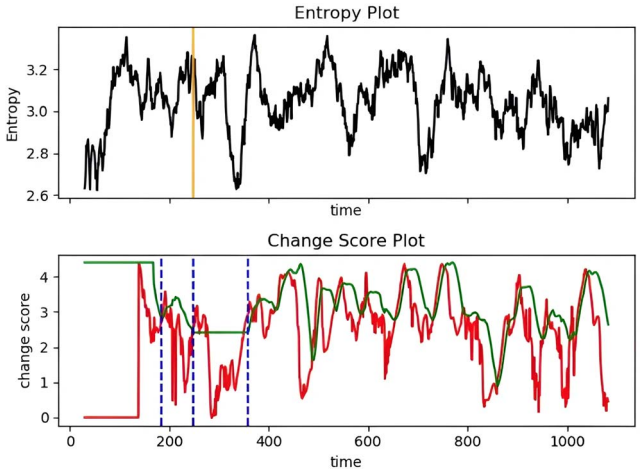

Figure 8. (Entropy vs time) and (change score vs time) plots for ARE abrupt stream using SIZE heuristic, taking window size as 30.

We now proceed to the second experiment in which we have applied the LSTM-based graph stream classification model (Figure 6) to the generated graph streams. Whenever classification is done taking drift into account, retraining of the model is done after each drift point detected by DSDD, effectively splitting the stream into two or more parts and running the model independently on each of them. Table 5 compares the performances obtained for each graph stream with and without incorporating drift detection using DSDD. Each model has been run five times for each stream, and the corresponding average and standard deviation of each performance metric are given. The same data is graphically represented by Figure 9. Here the yellow bars represent the accuracies obtained when drift detection has not been incorporated. The pink and purple bars correspond to the accuracies obtained when drift detection has been incorporated using MDL and SIZE heuristics, respectively. As is evident from streams such as Ahr-Abrupt and MMP-gradual, there has been substantial improvement in accuracy in most streams when drift detection has been incorporated, for both MDL and SIZE heuristics. Even in the case of precision and recall, all of the streams have seen very competitive results if not improved results when drift detection using DSDD has been used in comparison to that obtained without any drift detection. This enhancement in performance is no doubt partly due to the fast detection and high detection rate of DSDD. This improvement is clear if we compare the confusion matrices. Fig. 10a shows the confusion matrix obtained in one of the runs of the model when drift detection was not incorporated. Figures 10b and 10c showed the confusion matrices in one of the runs of the model when drift detection was incorporated using MDL and SIZE heuristics, respectively. These results clearly prove the importance of drift detection in toxicology graph streams and the effectiveness of DSDD in detecting drift.

Table 4. Gradual drift experiment results

Dataset	Window size	Heuristic	DDR	DoD	FA1000
Tox21-Ahr	30	MDL	**1.00 ± 0.00**	17.90 ± 18.48	**1.28 ± 0.02**
		SIZE	**1.00 ± 0.00**	**12.42 ± 18.44**	1.30 ± 0.09
	50	MDL	**1.00 ± 0.00**	**20.70 ± 21.14**	1.41 ± 0.29
		SIZE	**1.00 ± 0.00**	21.24 ± 21.28	**1.40 ± 0.31**
	150	MDL	0.88 ± 0.32	**29.36 ± 31.07**	2.21 ±0.60
		SIZE	**0.90 ± 0.30**	30.34 ± 33.17	**2.21 ± 0.58**
	300	MDL	**0.22 ± 0.41**	20.70 ± 40.82	1.57 ± 0.52
		SIZE	0.14 ± 0.35	**12.76 ± 33.59**	**1.51 ± 0.51**
Tox21-ARE	30	MDL	**1.00 ± 0.00**	**12.28 ± 15.90**	1.75 ± 0.33
		SIZE	**1.00 ± 0.00**	19.48 ± 21.41	**1.64 ± 0.42**
	50	MDL	**1.00 ± 0.00**	**25.62 ± 21.03**	**1.01 ± 0.28**
		SIZE	0.98 ± 0.14	28.10 ± 20.57	1.14 ± 0.39
	150	MDL	**0.02 ± 0.14**	1.76 ± 12.32	**1.86 ± 0.65**
		SIZE	**0.02 ± 0.14**	**1.28 ± 8.96**	2.10 ± 0.69
	300	MDL	**0.00 ± 0.00**	**0.00 ± 0.00**	**1.46 ± 0.59**
		SIZE	**0.00 ± 0.00**	**0.00 ± 0.00**	1.48 ± 0.64
Tox21-MMP	30	MDL	**1.00 ± 0.00**	**10.28 ± 14.44**	**1.01 ± 0.10**
		SIZE	**1.00 ± 0.00**	16.32 ± 15.66	1.02 ± 0.12
	50	MDL	**1.00 ± 0.00**	**21.00 ± 20.72**	1.17 ± 0.28
		SIZE	0.96 ± 0.20	22.72 ± 23.43	**1.10 ± 0.23**
	150	MDL	**0.94 ± 0.24**	30.34 ± 30.97	**2.27 ± 0.57**
		SIZE	**0.94 ± 0.24**	**26.34 ± 26.48**	2.35 ± 0.55
	300	MDL	0.90 ± 0.30	**26.88 ± 25.87**	**1.90 ± 0.41**
		SIZE	**0.94 ± 0.24**	31.60 ± 33.16	1.95 ± 0.41
Tox21-Aromatase	30	MDL	**1.00 ± 0.00**	13.68 ± 13.56	2.06 ± 0.24
		SIZE	**1.00 ± 0.00**	**11.68 ± 14.92**	**2.00 ± 0.00**
	50	MDL	**1.00 ± 0.00**	**20.30 ± 19.80**	2.14 ± 0.35
		SIZE	**1.00 ± 0.00**	24.72 ± 21.35	**2.06 ± 0.24**
	150	MDL	0.74 ± 0.44	48.08 ± 32.20	**1.80 ± 0.72**
		SIZE	**0.76 ± 0.43**	**44.40 ± 31.50**	1.92 ± 0.66
	300	MDL	0.00 ± 0.00	**0.00 ± 0.00**	1.42 ± 0.60
		SIZE	**0.04 ± 0.20**	3.90 ± 19.15	**1.40 ± 0.49**

Table 5. Graph stream classification performances obtained without incorporating drift detection (Original) and with incorporating drift detection using DSDD (using both MDL and SIZE heuristics)

Graph stream	Drift detection method	Accuracy ± SD	Precision ± SD	Recall ± SD
Tox21-Ahr Abrupt	Without DSDD	52.58 ± 2.73	49.89 ± 0.51	49.38 ± 2.66
	MDL	87.16 ± 0.36	44.72 ± 0.01	48.58 ± 0.20
	SIZE	87.47 ± 0.86	44.59 ± 0.04	48.91 ± 0.48
Tox21-Ahr Gradual	Without DSDD	66.98 ± 2.75	61.65 ± 3.52	60.09 ± 3.44
	MDL	71.56 ± 1.39	62.59 ± 2.08	60.63 ± 1.71
	SIZE	72.07± 0.74	63.51± 1.14	61.72 ± 1.09
Tox21-ARE Abrupt	Without DSDD	43.01 ± 2.19	50.03 ± 0.08	50.18 ± 0.44
	MDL	89.90 ± 0.40	89.90 ± 0.40	90.07 ± 0.41
	SIZE	88.18 ± 0.37	88.21 ± 0.36	88.49 ± 0.36
Tox21-ARE gradual	Without DSDD	80.73 ± 1.98	53.27 ± 2.69	54.60 ± 3.93
	MDL	74.86 ± 0.53	46.12 ± 0.93	46.98 ± 0.75
	SIZE	74.37 ± 0.36	50.30 ± 1.05	50.32 ± 1.06
Tox21-MMP Abrupt	Without DSDD	59.94 ± 3.92	49.03 ± 0.38	45.28 ± 1.80
	MDL	89.37 ± 0.82	89.50 ± 0.78	89.61 ± 0.80
	SIZE	88.42 ± 0.29	88.46 ± 0.29	88.69 ± 0.29
Tox21-MMP Gradual	Without DSDD	47.36 ± 3.18	51.16 ± 0.72	53.20 ± 1.96
	MDL	85.16 ± 1.11	85.11 ± 1.14	85.05 ± 0.98
	SIZE	80.94 ± 1.30	81.13 ± 1.23	81.28 ± 1.25
Tox21-aromatase Abrupt	Without DSDD	21.56 ± 4.78	49.66 ± 1.83	47.74 ± 4.46
	MDL	79.68 ± 0.75	80.26 ± 0.87	81.80 ± 0.89
	SIZE	78.60 ± 0.37	79.38 ± 0.41	81.35 ± 0.42
Tox21-aromatase Gradual	Without DSDD	55.59 ± 1.59	65.44 ± 3.41	55.59 ± 1.59
	MDL	70.54 ± 1.23	68.19 ± 1.47	66.31 ± 1.33
	SIZE	67.60 ± 1.28	64.60 ± 1.51	63.47 ± 1.47

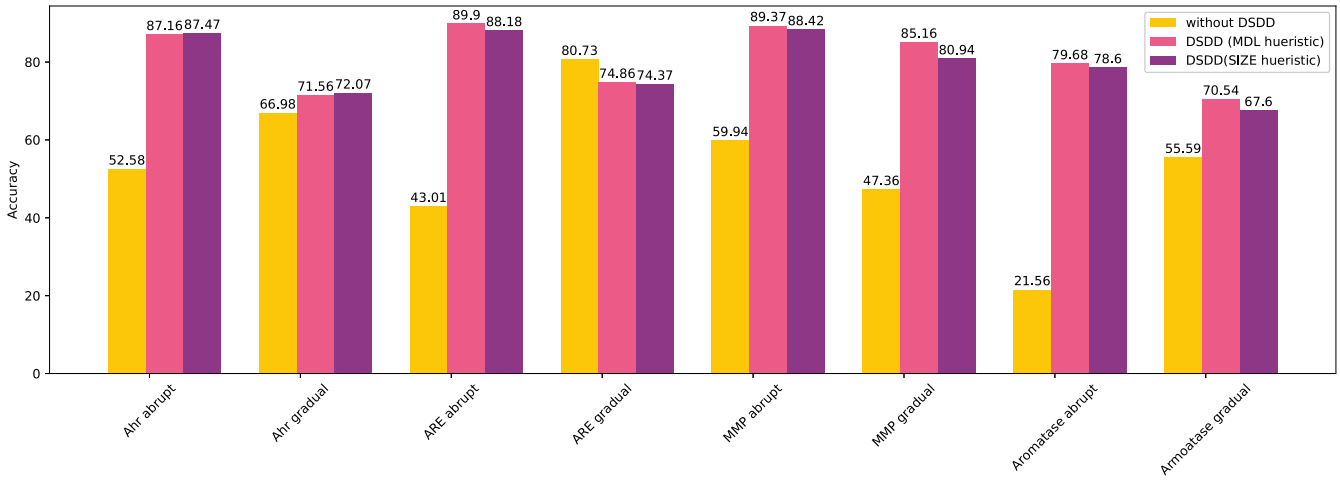

Figure 9. Graph stream classification accuracies obtained with and without using drift detection.

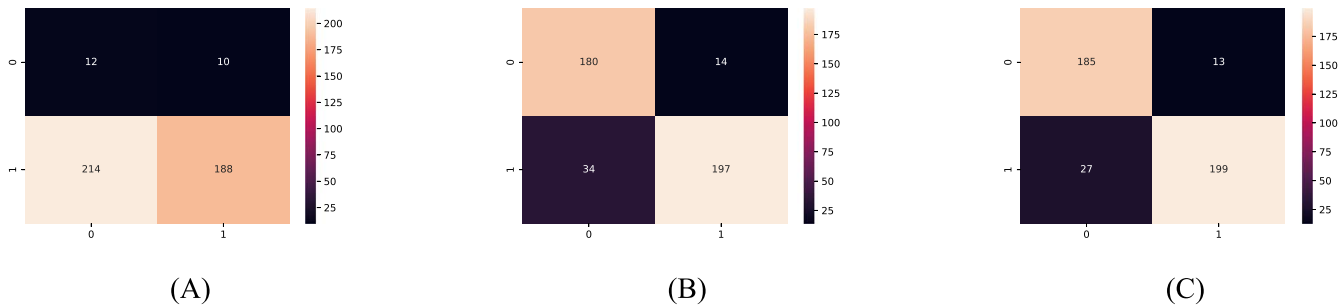

Figure 10. Confusion matrix for classification of ARE dataset in case of abrupt drift (**A**) when drift detection is not applied, (**B**) when drift detection is applied using MDL heuristic, (**C**) when drift detection is applied using SIZE heuristic.

There are a few minor exceptions to this general trend of substantial improvement after incorporating DSDD, such as in the case of ARE-gradual stream, in which the performance without drift detection is better than that obtained with drift detection, and in the case of Ahr-gradual stream, where there is only a slight improvement after using DSDD. These could be attributed to the fact that the graph stream classification performance depends not only on the data distribution but also on the data itself and the structural content of the molecules. Ahr gradual and ARE gradual streams have a slow change in distribution around the drift point. Slow changes are harder to detect and adapt when performing classification since there is no fixed point where the distribution changes. This could be partly responsible for the anomalous properties displayed by these two streams. This trend is quite apparent if we compare the abrupt and gradual streams. In the case of abrupt streams, the performance obtained after drift detection has been incorporated is quite substantial. However, when it comes to the gradual streams, the improvement is not too impressive, owing to the gradual nature of the drift point. ARE gradual, in addition, has an early drift point of 200, which could play a role in it being an exception.

Finally, it should also be noted that the performance is quite similar when MDL and SIZE are used. The accuracies, precisions, and recalls given by MDL are very similar to those given by SIZE for all the streams.

Conclusion and future scope

In this work, we have applied the DSDD to real-world toxicology datasets. Four datasets – Ahr, ARE, MMP and Aromatase were used, which were provided as part of the training datasets of the Tox21 Challenge. We have then simulated four graph streams with abrupt drift points and four graph streams with gradual drift points. These graph streams were generated with varying distributions and drift points. Extensive experiments have been performed by applying DSDD on graphs streams with abrupt drifts using MDL heuristic as well as SIZE heuristic with various window sizes – 30,50,150,300 and subsequently compared and analyzed the results obtained. Also similar experiments were performed on graph streams with gradual drifts in similar settings. Further, to analyze the effect of concept drift on graph stream classification, an LSTM-based graph stream classification model has been applied to the streams and the performance obtained with and without incorporating drift detection using DSDD, as well as adapting the model to the detected drift, has been compared. On the basis of experimental results, a number of observations have been noted. We have noticed that in graph streams, the performance mostly becomes better with decreasing window size for both abrupt and gradual drift streams. It is also observed that a simple heuristic such as SIZE (vertices + edges) provides a decent performance while at the same time being much simpler to calculate than MDL. Moreover, although MDL seems to perform better than SIZE generally, there are some cases where SIZE overtakes MDL in performance, both in the case of abrupt and gradual drift streams. We have also observed that DSDD performs quite well in toxicology graph streams with gradual drift, with the performance scores being on par with those of abrupt drift graph streams.

The usefulness of DSDD on these real-world graph streams was made clear when we applied the LSTM-based graph stream classification model to the streams. Most of the graph streams, regardless of having abrupt or gradual drifts, were observed to have a considerable improvement in classification accuracies when the drift was detected using DSDD and the model retrained accordingly, over when concept drift was ignored and not handled. Even in the case of precision and recall, the results obtained after drift detection was incorporated were found to be equally competitive, if not better, than when the drift was not handled. In the future, this work can be extended to different graph-based data modeling applications such as drug discovery. Multiclass classification problems can be another future direction.

Key Points

- Simulated graph streams containing both abrupt and gradual drift points from four real-life toxicology datasets and applied Discriminative Subgraph Based Drift Detector (DSDD) on them to detect drift points.
- Compared and analyzed the results obtained under different settings including varying window sizes and two different heuristics for discriminative subgraph extraction – MDL and SIZE.
- Performed graph stream classification on the graph streams containing drift points using a LSTM based graph stream classifier.
- Compared the graph stream classification scores with and without incorporating concept drift detection using DSDD and demonstrated the importance of drift detection in toxicology graph streams based on the superior results obtained when drift detection is incorporated.

Data availability

We used four datasets that have been provided as part of the Tox21 Data Challenge 2014 training datasets [22], which are publicly available at https://tripod.nih.gov/tox21/challenge/.

Acknowledgments

The support and resources provided by Centre for Computing and Information Services, IIT (BHU) Varanasi, are thankfully acknowledged.

References

1. Aggarwal CC. On classification of graph streams. In: *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM, 652–63.
2. Yao Y, Holder L. Scalable SVM-based classification in dynamic graphs. In: *2014 IEEE International Conference on Data Mining*. IEEE, 650–9.
3. Chi L, Li B, Zhu X. Fast graph stream classification using discriminative clique hashing. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 225–36.
4. Chi L, Li B, Zhu X, *et al.* Hashing for adaptive real-time graph stream classification with concept drifts. *IEEE Trans Cybernetics* 2017;**48**(5):1591–604.
5. Li B, Zhu X, Chi L, *et al.* Nested subtree hash kernels for large-scale graph classification over streams. In: *2012 IEEE 12th International Conference on Data Mining*. IEEE, 399–408.
6. Alippi C, Boracchi G, Roveri M. Just-in-time classifiers for recurrent concepts. *IEEE Trans Neural Networks Learn Syst* 2013; **24**(4):620–34.
7. Gama J, Medas P, Castillo G, *et al.* Learning with drift detection. In: *Brazilian Symposium on Artificial Intelligence*. Springer, 286–95.

8. Hulten G, Spencer L, Domingos P. Mining time-changing data streams. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*, Association for Computing Machinery, New York, NY, USA, 97–106.
9. Street WN, Kim Y. A streaming ensemble algorithm (SEA) for large-scale classification. In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDD '01)*, Association for Computing Machinery, New York, NY, USA, 377–82.
10. Yao Y, Holder LB. Detecting concept drift in classification over streaming graphs. In: *Proceedings of the KDD Workshop on Mining and Learning with Graphs (MLG'16)*, Association for Computing Machinery, New York, NY, USA, 2134–42.
11. Paudel R, Eberle W. An approach for concept drift detection in a graph stream using discriminative subgraphs. *ACM Trans Knowl Discov Data (TKDD)* 2020;**14**(6):1–25.
12. Zambon D, Alippi C, Livi L. Concept drift and anomaly detection in graph streams. *IEEE Trans Neural Netw Learn Syst* 2018; **29**(11):5592–605.
13. Mayr A, Klambauer G, Unterthiner T, *et al.* DeepTox: Toxicity prediction using deep learning. *Front Environ Sci* 2016;**3**:1–15.
14. Karim A, Mishra A, Newton MH, *et al.* Efficient toxicity prediction via simple features using shallow neural networks and decision trees. *ACS Omega* 2019;**4**(1):1874–88.
15. Rissanen J. *Stochastic Complexity in Statistical Inquiry*, Vol. 15. World scientific, 1998.
16. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput* 1997;**9**(8):1735–80.
17. Cook DJ, Holder LB. Substructure discovery using minimum description length and background knowledge. *J Artif Intell Res* 1993;**1**:231–55.
18. Graph-Based Anomaly Detection. https://users.csc.tntech.edu/weberle/gbad/index.html.
19. Dehmer M, Mowshowitz A. A history of graph entropy measures. *Inform Sci* 2011;**181**(1):57–78.
20. Shannon CE. A mathematical theory of communication. *Bell Syst Tech J* 1948;**27**(3):379–423.
21. Liu S, Yamada M, Collier N, *et al.* Change-point detection in time-series data by relative density-ratio estimation. *Neural Netw* 2013;**43**:72–83.
22. Huang R, Xia M, Nguyen D-T, *et al.* Tox21 challenge to build predictive models of nuclear receptor and stress response pathways as mediated by exposure to environmental chemicals and drugs. *Front Environ Sci* 2016;**3**(85): 1–9.
23. Bai Y, Ding H, Qiao Y, *et al.* Unsupervised inductive graph-level representation learning via graph-graph proximity. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*. AAAI Press, 1988–1994.