Interactive Process Drift Detection: A Framework for Visual Analysis of Process Drifts (Extended Abstract)

Denise Maria Vecino Sato
Graduate Program in Informatics
Pontifícia Universidade Católica do Paraná and Instituto Federal do Paraná
Curitiba, Brazil
0000-0003-1117-7082

Rafaela Mantovani Fontana
Department of Professional and Technological Education
Universidade Federal do Paraná
Curitiba, Brazil
0000-0001-6350-4167

Jean Paul Barddal
Graduate Program in Informatics
Pontifícia Universidade Católica do Paraná
Curitiba, Brazil
0000-0001-9928-854X

Edson Emilio Scalabrin
Graduate Program in Informatics
Pontifícia Universidade Católica do Paraná
Curitiba, Brazil
0000-0002-3918-179

***Abstract*—Interactive Process Drift Detection (IPDD) is a framework for visual analysis of process drifts. A process drift indicates a change in the process model occurred at some point in time. IPDD firstly generates process models for subparts of the event log using a sliding window approach. Then, it detects the drifts by evaluating similarity metrics calculated between adjacent process models; a difference in some of the metrics indicates a drift. The current implementation of IPDD generates the process models using the directly-follows graph (DFG) and applies two metrics: nodes and edges similarity. The user interface shows the drifts in the process models over time, allowing the user to visually understand the model changes. Also, the user can easily change the hyperparameters for the analysis and verify the results on the interface. The user interface of IPDD allows the user to evaluate the detected drifts by calculating the F-score metric, which is useful when using artificial datasets. The underlying idea is to ease the choice of a "good" value for the hyperparameter configuration, which is critical for almost any drift detection tool.**

Keywords— process drift detection, visual process analysis, process drift, concept drift

I. Introduction

Process mining aims at creating valuable knowledge about business processes obtained from information systems event data. Usually, process mining techniques assume the processes to be steady-state, i.e., the event data contains information from a unique version of the process. However, this assumption does not reflect the reality of the business processes, which constantly adapt to new regulations, improve performance, or enhance user experience. The situation where a process changes while being analyzed is named concept drift or process drift [1].

The change in the process can affect the ongoing instances, sudden or gradually. A sudden drift occurs when all the ongoing instances start to follow the new process model immediately. In a gradual drift, there is a period of time where instances from both versions of the process model coexist. The process drifts can also follow recurrent or incremental patterns. A recurrent drift indicates that a replaced process model can occur again. In an incremental drift, minor changes of the process model are implemented during some time. Sudden, gradual, incremental, and recurring are considered process drift types [2]. The process drift can also affect one or more perspectives of the process model. However, the most common perspective considered in the available tools is the control flow. Identifying and understanding the process drifts is relevant for business analysts because it improves their knowledge about the processes and enhances the quality of process mining analysis. Even when analysts perform offline process mining analysis, process drift detection can provide benefits, e.g., avoid complex discovered process models, improve conformance checking, or enhance processes based on their current state.

Different tools for detecting process drifts from event logs have been proposed, but the accuracy of the detection is usually related to the hyperparameter configuration [3]. The ProDrift plugin in Apromore [4], [5] and the ConceptDrift plugin in ProM [2] can detect different types of drifts (sudden and gradual); however, the focus is the change point and information about it. The user has to complement the drift analysis by executing a more exploratory mining slicing the event log based on the reported change points to understand the evolution of the process. A more recent tool, named VDD [6], detects the four types of drifts and allows the user to explore the drift using the process model. However, the tool is based on constraints mined over Declare models, and it mixes DFGs with the constraints to explain the dynamic of the process over time. None of the identified tools calculate an accuracy metric in the user interface.

Tunning the hyperparameter configuration to enhance the detection accuracy imposes a challenge to the proposed tools because the different approaches are affected by the hyperparameter configuration. IPDD aims to overcome this issue by providing an interactive user interface where the user quickly changes the parameter and visually evaluates the results. The tool provides visual process drift detection analysis by showing the distinct process models over time, in what we can consider a "replay" of the process models. IPDD also provides information about the differences against the previous model for each process model, enhancing the analysis. IPDD's current implementation detects sudden drifts in the control-flow perspective offline, which is a limitation.

II. IPDD Main Features

The IPDD framework detects the process drifts by analyzing the event log using a sliding window strategy. First, the user

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

XXX.X.XXXX.XXXX.X/XX/$XX.00 ©20XX IEEE

defines the window size based on the number of traces, and IPDD splits the log using tumbling windows. Then, it generates a model for each window and calculates the similarity metrics between adjacent models. The idea is to compare models mined from adjacent time slots using similarity metrics; when they are not similar, IPDD identifies a drift and characterizes the change based on the information provided by the metric.

The IPDD's current implementation mines the DFGs (process maps) from the traces in the time slots using the Pm4Py [7]. Then, the adjacent derived graphs are compared using the Nodes (NS) and Edges similarity (ES) metrics. NS is calculated using Eq. 1 [2], where n_p and n_q are the number of activities in the process maps P and Q (derived from adjacent windows) respectively, and n_{cs} indicates the number of common activities between P and Q. ES is calculated using Eq. 2, similar to NS: e_p is the number of edges in P, e_q is the number of edges in Q, and e_{cs} indicates the number of common edges in both P and Q.

$$NS = 2 * n_{cs} = (n_p + n_q) \quad (1)$$

$$ES = 2 * e_{cs} = (e_p + e_q) \quad (2)$$

IPDD calculates both metrics, and if one or both is less than 0, it marks the window as a drift. The F-score metric uses the True Positives (TP), False Positives (FP), and FN (False Negatives). A TP indicates a window reported as a drift containing a trace inputted as a real drift; an FP is counted when a window reporting a drift does not contain any trace informed as real drifts, and an FN is incremented when a window that does not report a drift contains any traces inputted as actual drifts.

Fig. 1 shows the tool's main screen, allowing users to easily change parameters and visually check the results. The parameter configuration panel is on top, where users must define the hyperparameter configuration before starting the analysis. After clicking on "Analyze Process Drifts", users can follow the current status in the "Status" area below the parameters panel. When the analysis finishes, IPDD shows the process drift analysis panel. There is a timeline of windows in the upper part of this panel, where users can click to inspect specific windows of the process model. The similarity metrics information (on the left side) is updated for each window selected, providing information about the differences between the current and the previous model. In the example, the ES indicates a drift that is characterized by two edges added. After IPDD finishes the analysis, the user can show the evaluation panel to calculate the F-score metric by clicking "Evaluate results". IPDD framework is described in more detail in [8]. Its source code is available in a public repository[1], the deployed application is available in a public node[2], and a demo video is available on YouTube[3].

III. Case Studies

Authors have proposed different tools for process drift detection. However, the methods are usually sensitive to the hyperparameter configuration. Moreover, almost all approaches apply windowing strategies – and defining a "good" value for the window size is still a challenge. Also, the adaptive approaches have some drawbacks; other parameters affect the detected drifts [3]. Our IPDD approach gives users the freedom to check different hyperparameter configurations to overcome this challenge visually.

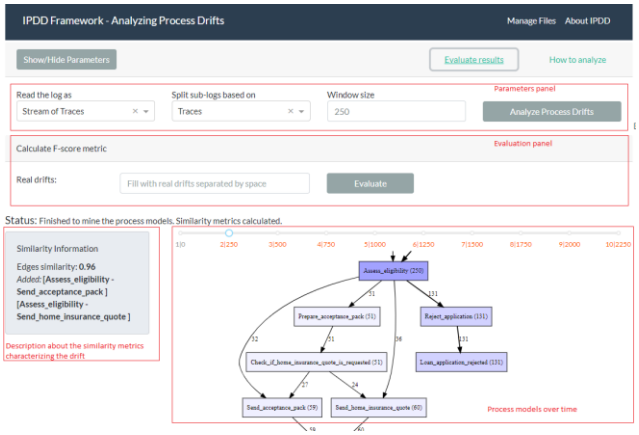

Fig.1. Screenshot from the main window.

The tool was presented to our research group in Curitiba (Brazil), including researchers from three post-graduate programs (Informatics, Production and Systems Engineering, and Health Technology). Firstly we have conducted a usability assessment for redesigning the user interface. Currently, we are working on a case study on a manufacturing scenario. The idea is to detect drifts in the temporal perspective of the process (sojourn time). The information about drifts will be used as input for planning the maintenance intervals on the production line.

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior-Brasil (CAPES) - Finance Code 001 – Grant No.: 88887.321450/2019-00.

References

[1] W. M. P. Van der Aalst et al., "Process Mining Manifesto," in International Conference on Business Process Management BPM 2011: Business Process Management Workshops, 2011, vol. 99, pp. 169–194.

[2] R. P. J. C. Bose, W. M. P. van der Aalst, I. Zliobaite, and M. Pechenizkiy, "Dealing With Concept Drifts in Process Mining," IEEE Trans. Neural Networks Learn. Syst., vol. 25, no. 1, pp. 154–171, Jan. 2014.

[3] S. M. Vecino, D. F. Cristiana, B. Paul, and S. Emilio, "A Survey on Concept Drift in Process Mining," ACM Comput. Surv., vol. 54, no. 9, pp. 1–38, Oct. 2021.

[4] A. Maaradji, M. Dumas, M. La Rosa, and A. Ostovar, "Fast and Accurate Business Process Drift Detection," in International Conference on Business Process Management BPM 2016: Business Process Management, 2015, pp. 406–422.

[5] A. Maaradji, M. Dumas, M. L. Rosa, and A. Ostovar, "Detecting Sudden and Gradual Drifts in Business Processes from Execution Traces," IEEE Trans. Knowl. Data Eng., vol. 29, no. 10, pp. 2140–2154, 2017.

[6] A. Yeshchenko, C. Di Ciccio, J. Mendling, and A. Polyvyanyy, "Visual Drift Detection for Sequence Data Analysis of Business Processes," IEEE Trans. Vis. Comput. Graph., pp. 1–1, 2021.

[7] A. Berti and S. van Zelst, "Process Mining for Python (PM4Py): Bridging the Gap Between Process- and Data Science." 2019.

[8] D. M. V. Sato, J. P. Barddal, and E. E. Scalabrin, "Interactive Process Drift Detection Framework," in International Conference on Artificial Intelligence and Soft Computing (ICAISC), 2021, pp. 192–204.

1 https://github.com/denisesato/InteractiveProcessDriftDetectionFW
2 http://visual-pro-drift.com.br:8050/
3 Demonstration video at: https://youtu.be/8feKd6jr8Gs