MODELLING MULTI-COMPONENT PREDICTIVE SYSTEMS AS PETRI NETS

Manuel Martin Salvador, Marcin Budka, Bogdan Gabrys
Bournemouth University
United Kingdom
{msalvador,mbudka,bgabrys}@bournemouth.ac.uk

KEYWORDS
Predictive Systems, Petri nets, Process industry

ABSTRACT

Building reliable data-driven predictive systems requires a considerable amount of human effort, especially in the data preparation and cleaning phase. In many application domains, multiple preprocessing steps need to be applied in sequence, constituting a 'workflow' and facilitating reproducibility. The concatenation of such workflow with a predictive model forms a Multi-Component Predictive System (MCPS). Automatic MCPS composition can speed up this process by taking the human out of the loop, at the cost of model transparency. In this paper, we adopt and suitably re-define the Well-handled with Regular Iterations Work Flow (WRI-WF) Petri nets to represent MCPSs. The use of such WRI-WF nets helps to increase the transparency of MCPSs required in industrial applications and make it possible to automatically verify the composed workflows. We also present our experience and results of applying this representation to model soft sensors in chemical production plants.

INTRODUCTION

In many data mining problems one needs to sequentially apply multiple preprocessing methods to the data (e.g. outlier detection → missing value imputation → dimensionality reduction), effectively forming a preprocessing chain. Such data-driven workflows have been used to guide data processing in a variety of fields. Some examples are astronomy (Berriman et al. 2007), biology (Shade and Teal 2015), clinical research (Teichmann et al. 2010), archive scanning (Messaoud et al. 2011), telecommunications (Maedche et al. 2000), banking (Wei et al. 2013) and process industry (Budka et al. 2014) to name a few. The common methodology in all these fields consists of following a number of steps to prepare a dataset for mining. In the field of predictive modelling, the workflow resulting from connecting different methods is known as a Multi-Component Predictive System (MCPS) (Tsakonas and Gabrys 2012). At the moment, tools like WEKA (Frank et al. 2016), RapidMiner (Hofmann and Klinkenberg 2016) or Knime (Berthold et al. 2008) allow to create and run MCPSs including a large variety of operators. Each of these tools however uses a different representation of workflows. One of the goals of this paper is to establish a common framework for connecting multiple data processing components into workflows.

ALTERNATIVES TO REPRESENT MCPS

To formalise the notion of MCPSs under a common abstract framework, various approaches can be considered:

1. **Function composition**, where each component is a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that makes an operation over an input tensor $\boldsymbol{x}$ (i.e. a multidimensional array made of continuous or categorical values) and returns an output tensor $\boldsymbol{y}$. Several components can be connected by composing functions i.e. $f(g(\boldsymbol{x}))$. However, this notation can become tedious when representing complex workflows involving multiple components with parallel paths of different lengths. Moreover, it is not expressive enough to represent different states of a concurrent system.

2. **Directed Acyclic Graphs (DAG)**, consisting of a set of components (nodes) F and directed arcs A connecting pairs of components: $G = (F, A)$. This approach is very flexible and makes it easy to understand the structure of the MCPS. However, DAGs on their own are not expressive enough to model system execution (e.g. how data is transformed, what metadata is generated, iterations, and preconditions) or temporal behaviour (e.g. duration and delays).

3. **Petri nets (PN)**, which are a modelling tool applicable to many types of systems (Petri 1962). A Petri net $PN = (P, T, F)$ is a directed bipartite graph consisting of a set of places P and transitions T connected by arcs F. Depending on the system, places and transitions can be interpreted in different ways. In this paper, places are considered to be data buffers and transitions are data processing methods. PNs have been shown to be very useful to model workflows (van der Aalst 1998a) since they are very flexible, can accommodate complex process logic including concurrency and have a strong mathematical foundation (Murata 1989). Using workflow

algebra (Pankratius and Stucky 2005) one can modify and create PNs with relational operators like *selection* or *union*, which can be useful to adapt workflows. Analysis methods like (van der Aalst 2000) inspect PN structure to find potential design errors. An important advantage is that the graphical nature of PNs makes them intuitive and easy to understand for any domain expert. PNs are also vendor independent and, once composed, can be easily translated to any data mining tool. This approach has not been considered before to model MCPSs and it is proposed in this paper for the first time.

BACKGROUND

While Petri nets were introduced in (Petri 1962), the most recent definition of a Petri net, which has been adopted in this paper, was given in (Murata 1989) as the following tuple

$$PN = (P, T, F, W, M_0) \tag{1}$$

where $P = \{p_1, ..., p_m\}$ is a finite set of places, $T = \{t_1, ..., t_n\}$ is a finite set of transitions, $F \in (P \times T) \cup (T \times P)$ is a set of arcs, $W : F \to \mathbb{N}^+$ is a weight function, $M_0 : P \to \mathbb{N}$ is the initial marking (i.e. state of the net – number of tokens in each place). Additionally, a Petri net contains one or more tokens that represent units of the system to be processed. The lifetime of a PN is defined by a set of states $\mathcal{M} = \{M_0, ..., M_q\}$. Each state is the distribution of the tokens over P.
In the remainder of this section, we define a number of properties of PNs, which enable us to apply this formalism to defining the MCPSs.
In a Petri net, nodes are connected by arcs forming paths. Formally,

Definition 1 (Path) *A path C from a node $n_1 \in P \cup T$ to a node $n_k \in P \cup T$ is a sequence of nodes $\langle n_1 n_2 ... n_k \rangle$ such that $f_{i,i+1} \in F \mid 1 \leq i \leq k-1$. A path C is elementary iff, for any two nodes n_i and n_j on C, $i \neq j \Rightarrow n_i \neq n_j$.*

The behaviour of a Petri net is described by firing of transitions. A transition can be fired (i.e. activated) when each of its input places are marked at least with the number of tokens indicated by the value of the function $w(p, t)$ associated with the arc $p \to t$ (i.e. minimum number of tokens needed in p to fire the transition t). When the active transitions t are fired, the state of the net changes from M_n to M_{n+1} and tokens are transferred from input to output places for each transition.
The nodes of a Petri net (i.e. places and transitions) can have multiple output and input arcs. Depending on the behaviour of the node, there are four main constructs:

- **AND-split**: A token is produced for each of the output arcs.
- **XOR-split**: A token is produced for only one of the output arcs.
- **AND-join**: A token is received for each of the input arcs.
- **XOR-join**: A token is received for only one of the input arcs.

The input nodes of a node $n \in P \cup T$ are denoted as $\bullet n$, while output ones are $n\bullet$.
Despite of the fact that the classical Petri nets can cover a large range of applications (e.g. manufacturing (DiCesare et al. 1993), business process management (van der Aalst et al. 2000), hardware design (Yakovlev et al. 2000), molecular biology systems (Hardy and Robillard 2004), there are sometimes circumstances when new properties have to be defined in order to cover additional types of systems. For example, (van der Aalst 1998a) presented a new type of Petri net to represent business process logic called WorkFlow net (WF-net). WF-nets are sequential workflows with a single starting point and ending point. The simplest WF-net is shown in Figure 1, where transitions represent tasks, places could be seen as conditions, and tokens are cases (e.g. a patient, a document or a picture). Formally:

Definition 2 (WF-net) *A Petri net is a WF-net iff:*

a) there is only one source place $i \in P \mid \bullet i = \emptyset$;

b) there is only one sink place $o \in P \mid o\bullet = \emptyset$;

c) every node $n \in P \cup T$ is on a path from i to o.

The third point of this definition entails that if a new transition t connecting o with i is added, then the resulting Petri net is strongly connected.

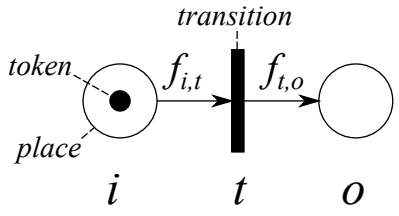

Figure 1: WorkFlow Net with a Single Transition

Definition 3 (Strong connectivity) *A Petri net is strongly connected iff for every pair of nodes (i.e. places and transitions) x and y, there is a path leading from x to y.*

The **soundness** property for WF-nets introduced by (van der Aalst 1997) implies that if a net has k tokens in the input place during the initial marking, it will have k tokens in the output place at the final marking (i.e. $M_0(i) = M_q(o)$). A WF-net with soundness is

guaranteed to terminate (i.e. it does not have deadlocks or livelocks).
In order to avoid deadlocks in WF-nets, (van der Aalst 2000) introduced the well-handled property that ensures the lack of bad constructions (e.g. XOR-split followed by AND-join block). Formally:

Definition 4 (Well-handledness) *A Petri net is well-handled iff, for any pair of nodes* $\{n_i, n_j\}$ *such that one is a place and the other a transition, and for any pair of elementary paths* C_1 *and* C_2 *leading from* n_i *to* n_j, $C_1 \cap C_2 = \{n_i, n_j\} \Rightarrow C_1 = C_2$.

Petri nets can become very large when defining complex processes (van der Aalst 1998b). To facilitate the representation, **hierarchical** Petri nets were introduced as an extension of PNs where a transition can be represented by another PN (called subnet) – see Figure 2. The action of replacing a transition by a subnet is called an iteration. Iterations are denoted as regular when the subnet has entrance and exit nodes acting as dummy nodes. This concept leads to a new type of Petri nets known as WRI-WF nets (Well-handled with Regular Iterations WF-net) presented in (Ping et al. 2004). Formally:

Definition 5 (WRI-WF net) *A Petri net is a WRI-WF net iff:*

a) the PN is a WF-net (see Definition 2);

b) the PN is well-handled (see Definition 4);

c) the PN is acyclic;

d) the iterations of the PN are regular.

WRI-WF nets are inherently sound (see (Ping et al. 2004) for proof).

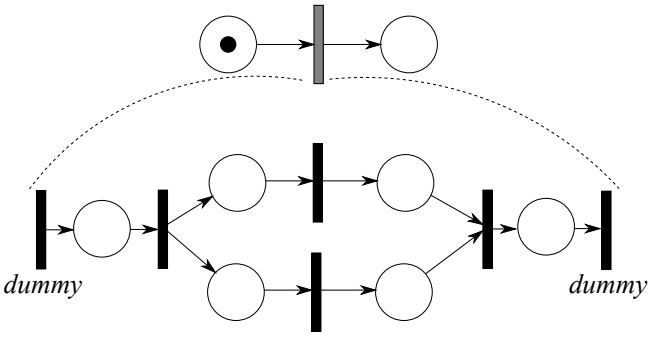

Figure 2: Hierarchical WF-net with Parallel Paths

MCPS DEFINITION

We propose to use WRI-WF-nets as the base for defining MCPSs. However, to comply with the predictive nature of MCPSs there are some additional restrictions that have to be added. Formally:

Definition 6 (MCPS) *A Petri net is an MCPS iff all the following conditions apply:*

a) the PN is WRI-WF-net (see Definition 5);

b) each place $p \in P \setminus \{i, o\}$ *has only a single input and a single output;*

c) the PN is 1-bounded, that is, there is a maximum of one token in each $p \in P$ *for every reachable state (i.e.* $M(p) \leq 1$*).;*

d) the PN is 1-sound (i.e. $M_0(i) = M_q(o) = 1$*);*

e) the PN is ordinary (i.e. $w = 1\ \forall w \in W$*);*

f) all the transitions $t \in T$ *with multiple inputs or outputs are AND-join or AND-split, respectively;*

g) any token is a tensor (i.e. multidimensional array).

In an MCPS, an atomic transition $t \in T$ is an algorithm with a set of hyperparameters λ that affect how the token is processed. An MCPS can be as simple as the one shown in Figure 1 with a single transition. For example, the token in i can be a set of unlabelled instances, and t a classifier which consumes such token from the arc $f_{i,t}$ and generates one token in o with the predicted labels through $f_{t,o}$.
An MCPS can however be hierarchically extended since each transition t can be either atomic or special (a subnet with additional starting and ending dummy transitions) – see Figure 2 where atomic transitions are black and special transitions are grey. As a consequence, an MCPS can model very complex systems with multiple data transformations and parallel paths (see e.g. Figure 5 for a multi-hierarchy example).
In predictive modelling, the semantics for transitions are: (1) preprocessing methods, (2) predictors, (3) ensembles and (4) postprocessing methods. Transitions representing (1), (2), and (4) can be either atomic or special. However, type (3) transitions are necessarily special since ensembles are made of several predictors and a combination method (e.g. voting).
Depending on the number of inputs and outputs, MCPSs can have any of the following types of transitions (see Figure 3):

- $1 \rightarrow 1$ transitions (e.g. a classifier that consumes unlabelled instances and returns predictions)
- $1 \rightarrow n$ transitions (e.g. a random subsampling method that consumes a set of instances and returns several subsets of data)
- $n \rightarrow 1$ transitions (e.g. a voting classifier that consumes multiple predictions per instance and returns a single prediction per instance)

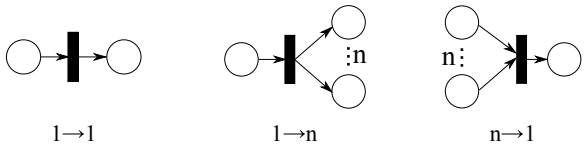

Figure 3: Types of transitions according to the number of inputs and outputs

APPLICATION IN PROCESS INDUSTRY

Processing plants have a large number of sensors that measure physical properties in different parts of the process. Values such as temperatures, pressures or humidity are easy to capture. However, acquiring other measurements is more expensive and often require human interaction. For instance, measuring the product concentration may require taking a sample and analysing it in the laboratory.

In order to improve production efficiency, a predictive model could deliver estimates of such hard-to-measure values based on the process state given by the easy-to-measure values from the sensors. This type of predictive models are called *soft sensors* because they can be seen as software or virtual sensors instead of physical.

Sometimes the first-principle models, that are based on the physical and chemical process knowledge, are available. Although such models are preferred by practitioners (De Assis and Maciel Filho 2000, Prasad et al. 2002), they are primarily meant for planning and design of the processing plants, and therefore usually focus on the steady states of the process (Chéruy 1997). Thus, such models can seldom be used in practice in a wide range of operating conditions. Moreover, often the process knowledge for modelling is not available at all. In such cases data-driven models fill the gap and often play an important role for the operation of the processes as they can extract the process knowledge automatically from the provided data. A review of data-driven soft sensors in the process industry is presented in (Kadlec et al. 2009). The most popular methods are multivariate statistical techniques like Principal Component Analysis (PCA) in a combination with a regression model (PCR) (Jolliffe 2002), and Partial Least Squares (PLS) (Wold et al. 2001). Other common approaches like Multi-Layer Perceptron (MLP) (Qin 1997) and Radial Basis Function (RBF) (Wang et al. 2006) are based on neural networks. (Kadlec et al. 2009) show that there are indeed dozens of methods to build soft sensors, each of them with various tunable hyperparameters. There is however no single method that is universally superior across all the problems (Wolpert and Macready 1997).

Although the most common application of soft sensors is online prediction, others include process monitoring, fault detection and sensor backup. In any of them, the main requirements in the process industry are:

- reliability – to provide truthful results;
- robustness – to work under any circumstances or inconvenience; and
- transparency – to be comprehensible by human experts.

Within a joint research project with a large chemical manufacturer we have worked on the development of data-driven soft-sensors. As a consequence, we developed guidelines for building MCPSs made of data preprocessing methods and predictive models (Budka et al. 2014). Nevertheless, designing and optimising a predictive system to work in a real environment requires a considerable human effort – e.g. (Martin Salvador et al. 2014). Reducing these labour-intensive tasks led us to investigate the automation of composing of MCPS (Martin Salvador et al. 2016b). However, the black-box nature of the automatic process reduces the transparency requirement from the industry. Representing MCPSs as Petri nets helped us to address such requirement.

In order to automatically compose and optimise MCPSs for a given classification or regression problem, we have adopted Auto-WEKA (Thornton et al. 2013), a tool for combined model selection and hyperparameter optimization. We have extended this software to generate a Petri net of the resultant MCPS in PNML (Petri Net Markup Language) format. This PN can then be analysed in any tool supporting this standard language (e.g. WoPeD (Freytag and Sänger 2014)). The software and source code are publicly available in our repository: `https://github.com/dsibournemouth/autoweka`.

EXPERIMENTAL RESULTS

We have carried out an experimental analysis of automatically composing and optimising MCPSs for 7 datasets representing regression tasks of real chemical processes. Four of these datasets have been made available by Evonik Industries as part of the collaboration within the INFER project (Musial et al. 2013), and have been extensively used in previous studies (Kadlec and Gabrys 2009, Budka et al. 2014, Bakirov et al. 2015):

- 'absorber' dataset which contains 38 continuous attributes from an absorption process. No additional information has been provided apart from this being a regression task;
- 'drier' dataset, which consists of 19 continuous features from physical sensors (i.e. temperature, pressure and humidity) and the target value is the residual humidity of the process product (Kadlec and Gabrys 2009);
- 'oxeno' dataset, which contains 71 continuous attributes also from physical sensors and a target vari-

able which is the product concentration measured in the laboratory (Budka et al. 2014); and

- 'thermalox' dataset, which has 38 attributes from physical sensors and the two target values are concentrations of NO_x and SO_x in the exhaust gases (Kadlec and Gabrys 2009).

Due to confidentiality reasons the datasets listed above cannot be published. However, 3 additional publicly available datasets from the same domain have also been used in the experiments. These are:

- 'catalyst' dataset consisting of 14 attributes, where the task is to predict the activity of a catalyst in a multi-tube reactor (Kadlec and Gabrys 2011);
- 'debutanizer' dataset, which has 7 attributes (temperature, pressure and flow measurements of a debutanizer column) and where the target value is the concentration of butane at the output of the column (Fortuna et al. 2005); and
- the 'sulfur' recovery unit dataset, which is a system for removing environmental pollutants from acid gas streams before they are released into the atmosphere (Fortuna et al. 2003). The washed out gases are transformed into sulfur. The dataset has five input features (flow measurements) and two target values: concentration of H_2S and SO_2.

Each dataset has been split into 70% training and 30% testing sets, unless partition was already provided. Auto-WEKA was run in parallel with 25 different initial seeds and taking the Root Mean Squared Error (RMSE) as optimisation measure. The best MCPS configurations for each dataset (i.e. lowest holdout error) are shown in Table 1. Ensemble methods (Bagging and RandomSubSpace), which train multiple predictors with different subsets of data, have been found to provide the best performance for all analysed datasets. In fact, the solutions found outperform the four most popular methods for building soft sensors (PCR, PLS, MLP and RBF) in 6 out of 7 datasets (see δ in Table 1). None of the most popular techniques have been selected among the best MCPSs, indicating the potential disadvantage of human bias towards well-known methods.
As a result of our new extension, we have been able to generate Petri nets like the ones shown in Figures 4 and 5, in which transitions represent the WEKA methods to process the datasets 'debutanizer' and 'sulfur', respectively.

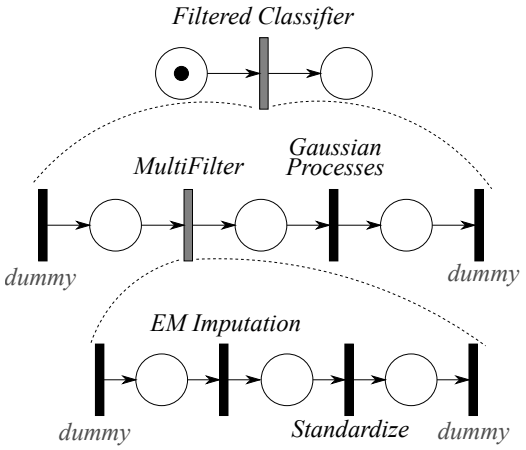

Figure 4: MCPS for 'debutanizer' Dataset

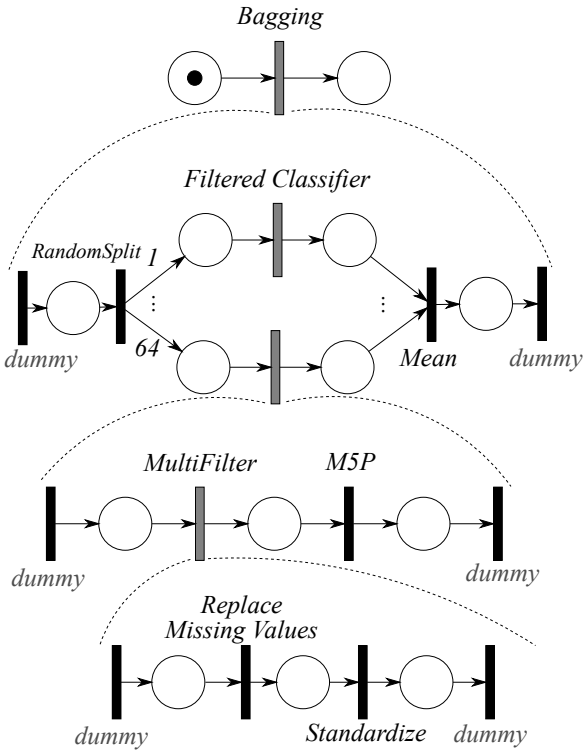

Figure 5: MCPS for 'sulfur' Dataset

CONCLUSION AND FUTURE WORK

In this paper we have proposed a novel definition of multi-component predictive systems (MCPSs) based on Petri nets. This vendor-independent formulation opens the door to formally verify that MCPSs are correctly composed (Sadiq et al. 2004), which is still an outstanding and non-trivial problem.

The experimental results show that it is feasible to automate the composition of MCPS for a real application such as process industry. The automatic generation of Petri nets helps to increase the transparency of the predictive system which a priori might be hidden under a black-box process. Moreover, since Petri nets are vendor independent, they can be easily translated to any data mining tool.

Petri nets are very useful to express semantic of complex

dataset	MV	TR	DR	SA	predictor	meta-predictor	$\mathcal{E}$	δ
absorber	-	Wavelet	Rand.Subs.	-	KStar	Rand.SubSp.	0.8989	↑ 0.0844
catalyst	Max	Normalize	-	-	GP	Bagging	0.0736	↑ 0.1144
debutanizer	EM	Wavelet	-	-	IBk	Rand.SubSp.	0.1745	-0.0035
drier	EM	-	Rand.Subs.	Res.Samp.	M5P	Bagging	1.3744	↑ 0.0573
oxeno	Zero	Normalize	-	-	M5P	Rand.SubSp.	0.0226	↑ 0.0042
sulfur	Zero	Standardize	-	-	M5P	Bagging	0.0366	↑ 0.0030
thermalox	Mean	Wavelet	-	-	GP	Rand.SubSp.	0.6904	↑ 0.6170

Table 1: Best MCPS for each dataset, holdout error $\mathcal{E}$ and difference with baseline δ (↑ indicates an improvement). MV = missing value replacement, TR = transformation, DR = dimensionality reduction, SA = sampling.

systems. For example, we have also explored the use of coloured Petri nets to include an additional token representing meta-data when transitions of the MCPS need adaptation (Martin Salvador et al. 2016a). As future work, we would like to take advantage of workflow algebra (Pankratius and Stucky 2005) to model the adaptation of MCPSs. Moreover, it would be interesting to use Timed Petri nets to model task duration and delays.

REFERENCES

Bakirov R.; Gabrys B.; and Fay D., 2015. *On sequences of different adaptive mechanisms in non-stationary regression problems.* In *Proc. of the International Joint Conf. on Neural Networks*. vol. July, 1–8.

Berriman G.B.; Deelman E.; Good J.; Jacob J.C.; Katz D.S.; Laity A.C.; Prince T.A.; Singh G.; and Su M.H., 2007. *Generating complex astronomy workflows*, Springer. 19–38.

Berthold M.; Cebron N.; Dill F.; and Gabriel T., 2008. *KNIME: The Konstanz information miner. Data Analysis, Machine Learning and Applications*, 1–8.

Budka M.; Eastwood M.; Gabrys B.; Kadlec P.; Martin Salvador M.; Schwan S.; Tsakonas A.; and Žliobaitė I., 2014. *From Sensor Readings to Predictions: On the Process of Developing Practical Soft Sensors.* In *Advances in IDA XIII*. vol. 8819, 49–60.

Chéruy A., 1997. *Software sensors in bioprocess engineering. Journal of Biotechnology*, 52, no. 3, 193–199.

De Assis A.J. and Maciel Filho R., 2000. *Soft sensors development for on-line bioreactor state estimation. Computers and Chemical Engineering*, 24, no. 2-7, 1099–1103.

DiCesare F.; Harhalakis G.; Proth J.M.; Silva M.; and Vernadat F.B., 1993. *Practice of Petri Nets in Manufacturing*. Springer.

Fortuna L.; Graziani S.; and Xibilia M.G., 2005. *Soft sensors for product quality monitoring in debutanizer distillation columns. Control Engineering Practice*, 13, 499–508.

Fortuna L.; Rizzo A.; Sinatra M.; and Xibilia M.G., 2003. *Soft analyzers for a sulfur recovery unit. Control Engineering Practice*, 11, 1491–1500.

Frank E.; Hall M.A.; and Witten I.H., 2016. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann.

Freytag T. and Sänger M., 2014. *WoPeD - An educational tool for workflow nets.* In *CEUR Workshop Proceedings*. vol. 1295, 31–35.

Hardy S. and Robillard P.N., 2004. *Modeling and simulation of molecular biology systems using petri nets: modeling goals of various approaches. Journal of Bioinformatics and Computational Biology*, 02, no. 04, 619–637.

Hofmann M. and Klinkenberg R., 2016. *RapidMiner: Data mining use cases and business analytics applications*. CRC Press.

Jolliffe I.T., 2002. *Principal Component Analysis, Second Edition*. Springer.

Kadlec P. and Gabrys B., 2009. *Architecture for development of adaptive on-line prediction models. Memetic Computing*, 1, no. 4, 241–269.

Kadlec P. and Gabrys B., 2011. *Local learning-based adaptive soft sensor for catalyst activation prediction. AIChE Journal*, 57, no. 5, 1288–1301.

Kadlec P.; Gabrys B.; and Strandt S., 2009. *Data-driven Soft Sensors in the process industry. Computers & Chemical Engineering*, 33, no. 4, 795–814.

Maedche A.; Hotho A.; and Wiese M., 2000. *Enhancing Preprocessing in Data-Intensive Domains using Online-Analytical Processing.* In *Second Int. Conf. DaWaK 2000*. 258–264.

Martin Salvador M.; Budka M.; and Gabrys B., 2016a. *Effects of change propagation resulting from adaptive preprocessing in multicomponent predictive systems.* In *20th International Conference on Knowledge Based and Intelligent Information and Engineering Systems*. vol. 96, 713–722.

Martin Salvador M.; Budka M.; and Gabrys B., 2016b. *Towards Automatic Composition of Multicomponent Predictive Systems*. In *11th International Conference on Hybrid Artificial Intelligence Systems*. 27–39.

Martin Salvador M.; Gabrys B.; and Žliobaitė I., 2014. *Online Detection of Shutdown Periods in Chemical Plants: A Case Study*. In *18th International Conference in Knowledge Based and Intelligent Information and Engineering Systems*. vol. 35, 580–588.

Messaoud I.; El Abed H.; Märgner V.; and Amiri H., 2011. *A design of a preprocessing framework for large database of historical documents*. In *Proc. of the 2011 Workshop on Historical Document Imaging and Processing*. 177–183.

Murata T., 1989. *Petri nets: Properties, analysis and applications*. 77, no. 4, 541–580.

Musial K.; Budka M.; and Blysz W., 2013. *Understanding the Other Side The Inside Story of the INFER Project*. *Smart Innovation, Systems and Technologies*, 18, 1–9.

Pankratius V. and Stucky W., 2005. *A Formal Foundation for Workflow Composition, Workflow View Definition, and Workflow Normalization based on Petri Nets*. In *2nd Asia-Pacific Conference on Conceptual Modelling (APCCM 2005)*. 79–88.

Petri C.A., 1962. *Kommunikation mit Automaten*. Ph.D. thesis, Universität Hamburg.

Ping L.; Hao H.; and Jian L., 2004. *On 1-soundness and Soundness of Workflow Nets*. In *Third Work. Model. Objects, Components, Agents*. 21–36.

Prasad V.; Schley M.; Russo L.P.; and Wayne Bequette B., 2002. *Product property and production rate control of styrene polymerization*. *Journal of Process Control*, 12, no. 3, 353–372.

Qin S., 1997. *Chapter 8 - Neural Networks for Intelligent Sensors and ControlPractical Issues and Some Solutions*. In *Neural Systems for Control*. 213–234.

Sadiq S.; Orlowska M.; Sadiq W.; and Foulger C., 2004. *Data Flow and Validation in Workflow Modelling*. In *Proc. of the 15th Australasian database conference*. vol. 27, 207–214.

Shade A. and Teal T.K., 2015. *Computing Workflows for Biologists: A Roadmap*. *PLoS Biol*, 13, no. 11, 1–10.

Teichmann E.; Demir E.; and Chaussalet T., 2010. *Data preparation for clinical data mining to identify patients at risk of readmission*. In *IEEE 23rd International Symposium on Computer-Based Medical Systems*. 184–189.

Thornton C.; Hutter F.; Hoos H.H.; and Leyton-Brown K., 2013. *Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms*. In *Proc. of the 19th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM Press, 847–855.

Tsakonas A. and Gabrys B., 2012. *GRADIENT: Grammar-driven genetic programming framework for building multi-component, hierarchical predictive systems*. *Expert Systems with Applications*, 39, no. 18, 13253–13266.

van der Aalst W.M.P., 1997. *Verification of workflow nets*. In *App. and Theory of Petri Nets*. 407–426.

van der Aalst W.M.P., 1998a. *The Application of Petri Nets To Workflow Management*. *Journal of Circuits, Systems and Computers*, 08, no. 01, 21–66.

van der Aalst W.M.P., 1998b. *Three Good Reasons for Using a Petri-Net-Based Workflow Management System*. In *Information and Process Integration in Enterprises*. 161–182.

van der Aalst W.M.P., 2000. *Workflow Verification: Finding Control-Flow Errors Using Petri-Net-Based Techniques*. *Business Process Management*, 1806, 19–128.

van der Aalst W.M.P.; Desel J.; and Oberweis A. (Eds.), 2000. *Business Process Management*, *Lecture Notes in Computer Science*, vol. 1806. Springer.

Wang L.; Shao C.; Wang H.; and Wu H., 2006. *Radial Basis Function Neural Networks-Based Modeling of the Membrane Separation Process: Hydrogen Recovery from Refinery Gases*. *Journal of Natural Gas Chemistry*, 15, no. 3, 230–234.

Wei W.; Li J.; Cao L.; Ou Y.; and Chen J., 2013. *Effective detection of sophisticated online banking fraud on extremely imbalanced data*. *World Wide Web*, 16, no. 4, 449–475.

Wold S.; Sjöström M.; and Eriksson L., 2001. *PLS-regression: A basic tool of chemometrics*. *Chemometrics and Intelligent Laboratory Systems*, 58, no. 2, 109–130.

Wolpert D.H. and Macready W.G., 1997. *No free lunch theorems for optimization*. *IEEE Transactions on Evolutionary Computation*, 1, no. 1, 67–82.

Yakovlev A.; Gomes L.; and Lavagno L. (Eds.), 2000. *Hardware Design and Petri Nets*. Springer.