Visual Analytics: A New Paradigm for Process Monitoring

Ibrahim Yousef * **Sirish L. Shah** ** **R. Bhushan Gopaluni** *

* *Department of Chemical and Biological Engineering, University of British Columbia, Vancouver, Canada (e-mail: iy641@mail.ubc.ca, bhushan.gopaluni@ubc.ca).*
** *Department of Chemical and Materials Engineering, University of Alberta, Edmonton, Canada (e-mail: sirish.shah@ualberta.ca)*

Abstract: As a result of recent breakthroughs in computer vision technologies, significant research interest has emerged to encode process data into visual clues and treat process monitoring problems as computer vision tasks. Imaging time-series signals as a feature engineering step forms a new branch of data analytics called "visual analytics". In the context of process monitoring, we define visual analytics as the integration of visual representation of the data combined with the use of computer vision tools and analytical reasoning to support decision-making and knowledge extraction from the data. In this work, a novel end-to-end visual analytics pipeline for industrial process fault detection using 1D and 2D convolution operations is proposed. The proposed approach presents a visual representation of data that captures temporal and local features from historical time-series signals. Next, the learned features in a 2D format are visually recognized and classified using 2D convolution operations. Our experimental results demonstrate that this approach achieves better performance on an industrial multivariate dataset compared to other state-of-art signals imaging tools such as Gramian Angular Field (GAF) and Recurrence Plots (RP).

Keywords: Process Monitoring, Signal Processing, Deep Learning, Image Analysis, Fault Detection, Vision, Applications.

1. INTRODUCTION

Driven by the advances in computational hardware and the availability of large volumes of industrial data, modern industry is witnessing a rapid transition from the classical manufacturing industry to the smart manufacturing industry (Reis and Gins (2017)). With the new drive towards Industry 4.0, process data analytics have become increasingly vital to solve longstanding problems involving process monitoring and control (Shang and You (2019)). Despite all the success that classical data analytics approaches (e.g. support vector machines, random forests, etc.) have seen in process monitoring tasks in recent years, there are still many practical limitations that need to be further refined and improved especially when dealing with complex industrial process data. First, most of the classical process data analytics frameworks separate the feature selection, extraction, and prediction steps which restricts their performance and makes the asynchronous optimization time-consuming (Jiao et al. (2019)). Furthermore, traditional methods usually seek to learn global features and ignore the temporal and local correlations that exist in process data (Yuan et al. (2020)).

Recently, deep learning (DL) models have been able to break into and improve various problem domains such as computer vision (CV), natural language processing, and facial and voice recognition. Unlike traditional data analytics techniques, DL models have a robust capability of learning high-level representations of the data and nonlinear patterns, assuming enough training data is available. Hence, significant research interest has been devoted to developing DL-based solutions for a variety of longstanding problems in many scientific and engineering disciplines. One of the most methodologically mature DL architectures used in CV is the 2D convolutional neural network (2D-CNN). Ever since winning the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012, 2D-CNN has become the dominant approach used in various CV tasks including image classification and object detection, mainly due to its ability to automatically learn local features from the raw input data. In addition, 2D-CNN consists of locally connected layers where only a subset of the input data is connected to each neuron. making them computationally efficient. Motivated by its astounding results in image recognition and its attractive attributes, the use of 2D-CNN for process systems engineering (PSE) applications is now a highly active research area.

Chemical manufacturing industries routinely generate a large volume of data, mostly in the form of time-series signals. A time-series signal $X_i = \{x_{i1}, x_{i2}, ..., x_{iN}\}$ is a sequence of N chronologically ordered observations with their accompanying timestamps $T = \{t_1, t_2, .., t_N\}$. When there are d different co-existing time-series signals recorded simultaneously by a set of d sensors, the data is referred to as a multi-variate time-series signal (MTS)

$M = (T, X_1, X_2, ..., X_d)$ with $X_i \in \mathbb{R}^N$. Note that we drop the time index from the MTS definition as T is unified across $X_1, X_2, .., X_d$. The difference in the intrinsic attributes between MTS data and image data, the typical 2D-CNN input data type, explains why 2D-CNN has not been adopted for time-series analysis until recently. For example, images are often described by their spatial information whilst MTS data are characterized by features related to time (autocorrelation) and observed variables (cross-correlations). Moreover, images are 3D tensors (i.e., $height \times width \times channels$) whereas MTS data are a collection of d 1D vectors (i.e., d column vectors of size $length \times 1$). Recently, many research studies have been dedicated to applying 2D-CNN on MTS data to solve process monitoring tasks. For example, 2D-CNN can be applied on raw MTS by directly re-arranging d column vectors of size N into a $N \times d$ dimensional matrix, where d represents the number of variables and N corresponds to the sample time length (Xia et al. (2018); Wu and Zhao (2018)). Additionally, Zheng et al. (2014) proposed a multi-channel CNN that extracts features from each univariate time-series separately using different CNNs, then the learned feature maps are concatenated and fed into a new CNN workflow. Hoang and Kang converted raw 1D vibration signals into gray-scale images called vibration images, then a 2D-CNN model with two hidden layers was constructed on the vibrated images for rolling bearing fault classification (Hoang and Kang (2019)). The latter approach is perhaps the most similar approach to the paradigm of visual analysis. Visual analytics was first defined as the science of analytical reasoning facilitated by an interactive visual interface (Cook and Thomas (2005)). However, in the context of process monitoring, we define visual analytics as the integration of visual representation of process data, the application of computer vision tools, and analytical reasoning to support decision-making and knowledge extraction from data as shown in Fig. 1

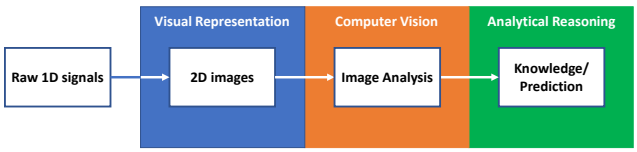

Fig. 1. *The process of visual analytics*

In this work, we propose a novel end-to-end visual analytics pipeline for industrial process fault detection using 1D and 2D convolution operations. The proposed approach obtains a visual representation of the data that captures the temporal and local features from historical MTS that would otherwise be spread over time. Next, the learned features in a 2D format are visually recognized and classified using a 2D-CNN. This work is a small step in the direction of migrating the techniques from computer vision, deep learning, and visual analytics to process industries. In addition, this work presents a promising opportunity to initiate the field of visual analytics for process monitoring problems. The main contributions of this paper include:

- The development of an end-to-end visual analytics pipeline that extracts discriminative features related to time and process variables in series.
- The introduction of a new visual representation learning method to encode MTS data into 2D images for manual image analysis.
- A comprehensive comparison between the proposed approach and other state-of-the-art CNN-based MTS classifiers on large-scale process data.

The remainder of this paper is organized as follows: In Section 2, we highlight the benefits of visual analytics as a process monitoring paradigm; Section 3 provides a theoretical background and presents related surveys and techniques; Section 4 describes our novel proposed visual analytics framework for fault detection tasks; the proposed methodology with other existing imaging tools are applied to an industrial case study introduced in Section 5; Section 6 outlines the experimental results followed by concluding remarks and prospects in Section 7.

2. WHY VISUAL ANALYTICS?

The recent advent of the Internet of Things (IoTs) has allowed the chemical manufacturing industries to routinely capture and access large data streams. This ever-increasing amount of data has triggered a major challenge towards extracting knowledge and discovering hidden opportunities contained therein. Indeed, this has led to the so-called "data deluge" problem. To address this data deluge, new technologies and frameworks have been developed to extract valuable and reliable information from unexplored process data.

To deal with information overload, chemical process operators often rely on the graphical representation of data "coordinates-based" for detecting anomalies and uncovering patterns. Process data visualization focuses on creating visual views and valuable renderings of process data. This approach is challenging in complex chemical processes where the state of the process is typically represented by a large number of variables. In such cases, dimensionality reduction tools such as principal component analysis (PCA) and partial least squares (PLS) could be useful to transform the high dimensional process data into 2D or 3D data that can be visualized easier. However, such tools are not often implemented in practice because the latent variables obtained don't provide any physical insights concerning the process conditions (i.e., have no physical meaning). Also, most of the current visualization methods are inadequate to handle industrial-scale process data.

On the other hand, visual analytics provide visual representations "pixel-based" of data that could offer valuable information and hints about process conditions. The images obtained using visual analytics tools allow process operators to manually analyze the patterns and texture contained therein which is easier to interpret and identify compared to analyzing time-series signals in the time domain. In addition, process operators could use visual analytics to develop intuition in relating different patterns in the images to different process operating modes. For instance, certain observations in the images (e.g., vertical lines, horizontal lines, single dots, etc.) could have different qualitative interpretations about the process condition (e.g., smooth or faulty operation) and behavior (e.g., periodic, stationary, etc.). Developing a good level of correlations or relations of the patterns in the visual rep-

resentation of the data to the process operating conditions could support model predictions (i.e., enhance the model interpretability). Another advantage of the visual analytics paradigm is that it produces 2D images that preserve the temporal dependency in MTS data for subsequent image analysis using CV tools such as 2D-CNN. However, it is worth noting that the obtained 2D images don't always capture all process-related information, and they exhibit some patterns that are not easily interpretable.

3. BACKGROUND & RELATED WORK

In this section, we first introduce a theoretical background on 1D and 2D convolution operations for ease of understanding. Then, we present related techniques and works.

3.1 1D convolution operations

1D convolution is the key operation used in 1D-CNNs which have recently become the dominant model for various signal processing applications (e.g., anomaly detection, structural health monitoring, etc.). In the context of 1D-CNN, a 1D convolution is a discrete linear operation performed between a p-channel vector of length L_{in} which can be defined as $X_i \in \mathbb{R}^{L_{in}}, i \in \{1,..,p\}$ and a collection of q 1D kernels of length L_W per input channel, expressed as $W_{(i,j)} \in \mathbb{R}^{L_W}$ with $i \in \{1,..,p\}$and$j \in \{1,..,q\}$, yielding a new output vector of length L_{out} with as many channels as the number of kernels; $Z_j \in \mathbb{R}^{L_{out}}$ for $j \in \{1,..,q\}$. Mathematically speaking, 1D convolution operation can be expressed as:

$$\begin{aligned} Z_j &= \sum_{i=1}^{p} \text{Conv1D}(X_i, W_{(i,j)}) \\ Z_j\ [k] &= \sum_{i=1}^{p} \sum_{l=1}^{L_W} X_i[k+l-1] W_{(i,j)}[l] \end{aligned} \tag{1}$$

We see that in 1D convolution, we slide a unique 1D kernel over an input channel, and multiply the overlapping values of the kernel and the input channel together for all $l \in L_W$ and $i \in p$. Note that the number of elements traversed per slide is defined by the stride s. The stride s affects both the computational complexity of the convolution operation and the size of the output vector L_{out}. In other words, the greater the s is, the less number of computational operations involved, and the smaller L_{out} will be. Hence, it is viewed as a hyper-parameter that balances the trade-off between memory usage and the loss of information. Fig. 2a and Fig. 2b illustrate the difference between 1D convolution with $s = 1$ and $s = 2$.

As shown in Fig. 2a and Fig. 2b, $L_{out} < L_{in}$ and that is because the convolution operation is undefined close to the input boundaries (i.e., the kernel lies outside the length of the input). This type of convolution is called *valid* convolution. Next, *same* convolution is the other type of convolution where the input vector is artificially "padded" by zero elements; producing an output of the same size as the input (i.e., $L_{in} = L_{out}$. Fig. 2c demonstrates the *same* convolution operation. The size of the output vector of 1D convolution can be computed using Equation 2 :

$$L_{out} = \frac{L_{in} + 2P - L_W}{s} + 1 \tag{2}$$

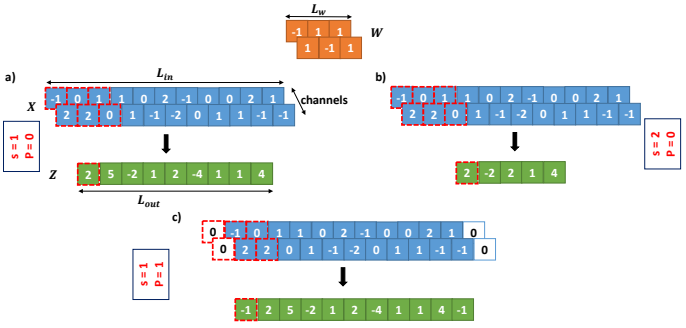

Fig. 2. *An illustration of 1D convolution with : a) stride = 1 and no padding; b) stride = 2 and no padding; c) stride = 1 and padding.*

P is defined as the padding. When *valid* convolution is performed, $P = 0$. On the other hand, in *same* 1D convolution, $L_{out} = L_{in}$ (i.e., $P = \frac{L_{in}(s-1)+L_W-s}{2}$).

3.2 2D convolution operations

Convolution operations in 2D are analogous to 1D convolution operations, and concepts introduced in the previous subsection (e.g., padding, stride, output size, etc.) still hold in 2D. The main difference between 1D and 2D convolution is that in 2D convolution, matrices substitute vectors for kernels and input data. In 2D convolution, we convolve a p-channel matrix of height H_{in} and width L_{in} $X_i \in \mathbb{R}^{H_{in} \times L_{in}}$ for $i \in \{1,..,p\}$ with a collection of q 2D kernels per input channel $W_{(i,j)} \in \mathbb{R}^{m \times m}$, $i \in \{1,..,p\}$and$j \in \{1,..,q\}$. This results in a new matrix $Z_j \in \mathbb{R}^{H_{out} L_{out}}$ for $j \in \{1,...,q\}$. Note, we assume that the kernels are square matrices with size $(m \times m)$. The *valid* 2D convolution operation can be defined as follows:

$$\begin{aligned} Z_j &= \sum_{i=1}^{p} \text{Conv2D}(X_i, W_{(i,j)}) \\ Z_j[k,d] &= \sum_{i=1}^{p} \sum_{l_2=1}^{m} \sum_{l_1=1}^{m} X_i[k+l_1-1, d+l_2-1] W_{(i,j)}[l_1, l_2] \end{aligned} \tag{3}$$

The computational complexity of 2D convolution between a filter $W \in \mathbb{R}^{m \times m}$ and an input matrix $X \in \mathbb{R}^{H_{in} \times L_{in}}$ is $\sim (m^2 H_{in} L_{in})$; consequently, 2D convolution becomes computationally expensive when the size of both the kernel W and the input matrix X are big; therefore, W is often of low dimension (i.e., $m << L_{in}, H_{in}$). Next, one can think of 2D convolution as a template matching task as shown in Fig. 3.

3.3 Imaging time-series signals

Time-series features are not always evident in the time domain. Hence, representing the temporal features of time-series signals as visual clues to expose hidden patterns and structures in the data has attracted much research interest in areas like signal processing and computer science. Encoding 1D time-series signals into 2D images enables subsequent analysis using 2D-CNN. In this study, we consider two of the most prominent time-series imaging tools which are: i) Gramian Angular Field (GAF) and ii) Recurrence Plot (RP).

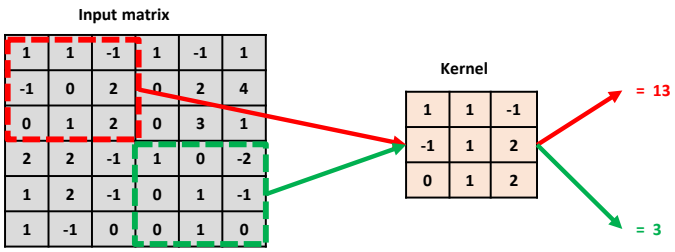

Fig. 3. *An illustration of 2D convolution. When the input patch matches the kernel, the convolution outputs a large activation.*

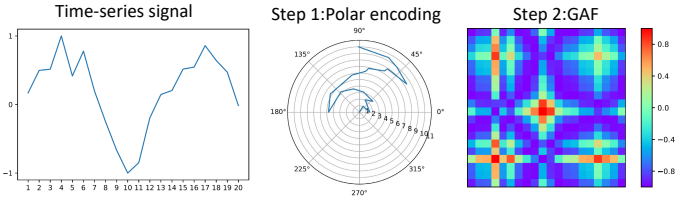

Fig. 4. *The two-steps instructions for constructing a 2D GAF image from a raw 1D time-series signal.*

A GAF image is a visual representation of a 1D time-series, introduced by Wang and Oates, that depicts information about the static behavior of the raw 1D time-series (Wang and Oates (2015)). In GAF images, 1D time-series signals are represented in polar coordinates (radius $\times$ angle) rather than the typical space coordinate system (time $\times$ magnitude). The process of encoding univariate time-series into 2D GAF images consists of two steps. First, the time-series is transformed from the space to polar coordinates by encoding the time step t_i as the radius r and the scaled time-series value x as the angular cosine θ, i.e.:

$$\begin{aligned} r_i &= \frac{t_i}{L}; & i \in N \\ \theta_i &= \cos^{-1}(x_i); & x_i \in [-1, 1] \end{aligned} \tag{4}$$

where L is a scaling constant and N is the total number of samples. Note that GAF is applied on scaled time-series (i.e., all time-series values fall in the interval [-1, 1]) to ensure a unique result in the polar coordinate system. This bijective property (i.e., one-to-one correspondence) is due to the monotonic nature of $\cos^{-1}(x)$ when $x \in [-1, 1]$. Once the scaled time-series signals are transformed into the polar coordinate system, the GAF matrix is constructed. In the GAF matrix, each entry denotes the cosine of the summation of angles, i.e.:

$$\mathrm{GAF}[i, j] = \cos(\theta_i + \theta_j); \qquad i, j = 1, 2, ..., N \tag{5}$$

Each pixel in the GAF image denotes the summation of the directions of two time-steps. The temporal dependency in the time-series is preserved in GAF images as time increases as the position goes from top-left to bottom-right. Fig. 4 illustrates the step-by-step instructions for generating GAF images from univariate time-series signals. The left side of Fig. 4 is a simple example of a scaled 1D time-series $x = (x_1, x_2, ..., x_{20})$ with $N = 20$ time steps. Then, x is transformed into a polar coordinate system using (4). Finally, the GAF image, a square matrix of size 20×20, is constructed using (5).

Dynamic non-linear processes often can be characterized by their recurrent behavior (e.g., periodicities, oscillations,

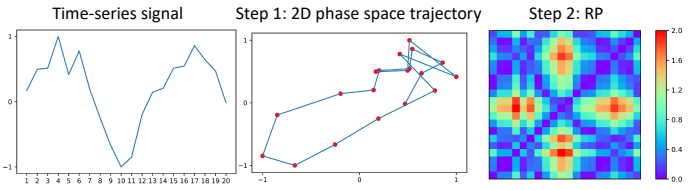

Fig. 5. *An illustration of the encoding map from a raw 1D time-series signal to a 2D RP image.*

irregular cyclicities) that are often not easy to visualize in the time domain. To tackle this challenge, Eckmann et al. introduced RP, a 2D visual representation of higher dimensional phase space trajectories (Eckmann et al. (1987)). RP is a square matrix that reveals at which points, the m-dimensional phase space trajectory revisits a previously visited state. In this work, we consider the non-binarized version of RP, as proposed by (Souza et al. (2014)), to avoid information loss when the matrix is binarized. In practice, one performs two steps to obtain an RP image from a univariate time-series. First, an embedding dimension m is chosen. The m-dimensional phase space state $\overrightarrow{s}$ is constructed from the time-series $x = (x_1, x_2, ..., x_N)$ using the time-series embedding (i.e., $\overrightarrow{s_i} = (x_i, x_{i+1}, ..., x_{i+m-1})$. Next, the RP matrix is calculated as follows:

$$\mathrm{RP}[i, j] = \|\overrightarrow{s_i} - \overrightarrow{s_j}\|; \qquad i, j = 1, 2, ..., K \tag{6}$$

where K is the total number of considered states $\overrightarrow{s}$ ($K = N - m + 1$) and $\|.\|$ is the Euclidean norm. Each pixel in RP denotes the Euclidean distance of two states in the m-dimensional phase space. RP exhibits texture (i.e., local patterns) and typology (i.e., global patterns) that offer information about the recurrent behavior of the 1D time-series. For instance, a periodic pattern in RP means that the process is cyclic. Moreover, fading to the top-left and bottom-right corners in RP indicates a non-stationary process. The full pipeline for calculating RP images from 1D time-series signal is shown in Fig. 5. The m-dimensional phase space trajectory is constructed from 1D signal x using the time delay embedding. In this examples, m is chosen as 2; hence, the states, represented in red dots, $\overrightarrow{s_i} = (x_i, x_{i+1})$. Next, the RP image, 19 × 19 matrix, is computed using (6). The colors in the RP image indicate the closeness of the states in the 2D phase space according to the corresponding color bar.

Recent studies have demonstrated that encoding raw 1D time-series signals as 2D images that are amenable to further 2D-CNN analytics provides competitive results (Garcia et al. (2021)). Hatami et al. converted raw 1D times-series into 2D texture RP images, then a deep 2D-CNN was constructed for time-series classification (Hatami et al. (2017)). The experimental results have shown that using RP images with 2D- CNN produces better results than other traditional time-series classifiers.

The main difference between the previous works and our work is that they applied some pre-defined rules or fixed functions on raw 1D time-series to display the visual representation of the data. While our proposed framework is more task-oriented, it learns the best visual representation of the raw time-series that minimizes the classification error of the network. Our results demonstrate that the joint learning of visual representation, visual

feature maps, and classifier parameters offered by the proposed framework could provide competitive results for fault detection on industrial operating data.

4. PROPOSED APPROACH

This section introduces a novel visual analytics pipeline for solving fault detection tasks. The proposed framework fuses data visual representation, visual feature extraction, and feature classification processes into a single learning problem. This network can effectively learn both the process dynamics and the various local correlations between different process variables during the training phase directly from the raw MTS signals. The overall architecture of the proposed visual analytics framework is shown in Fig. 6. This model consists of four main components: visual representation learning module, image formation module, visual feature extraction module, and a classification module. The four modules are introduced in the following subsections.

4.1 Visual representation learning module

The visual representation learning module takes in MTS signals. of any size, represented as p-channel vectors $X_i \in \mathbb{R}^{L_{in}}, i \in \{1, .., p\}$ where p is the number of input variables and L_{in} is the size/ length of the input signals (we assume that all input signals have the same length). The main objective of the module is to extract key information relative to time from the input MTS signals. This module is modeled by 1D convolutional layers, batch normalization layers, and non-linear activation functions.

In each 1D convolutional layer, 1D convolution operations are carried out between the input 1D data and q 1D kernels W. The size and the number of 1D kernels are hyper-parameters that are tuned to specify the "interesting" time interval size and the number of abstractions to be extracted from the input data, respectively. Each 1D convolutional layer produces a 1D feature map (i.e., output vector) per kernel. Disparate 1D kernels have different frequency content; thus, extracting different features from the input data. One can interpret a 1D feature map as the response of the input signal to the corresponding 1D kernel. Note that pooling layers in this work are discarded in favor of tuning strides s and padding P hyper-parameters in the convolution layers. Replacing pooling layers with convolution layers with larger strides has been found helpful in training generative models and it results in no degradation in model performance (Springenberg et al. (2015)).

Next, batch normalization layers ensure that the inputs to the hidden layers follow the same distribution (i.e., have normalized mean and variance) which enables a higher convergence rate of the network. Then, the non-linear activation function induces non-linearity in the network and allow the stacking of multiple layers required to learn complex patterns from the input data. This module produces a set of 1D feature maps to be fed and processed by the next module in the pipeline.

4.2 Image formation module

In this module, a 2D matrix is formed by vertically stacking the set of 1D feature maps extracted from the last layer L_1 of the previous module. The 2D matrix represents a collection of optimal 1D feature maps that best explains the temporal features of the input MTS signals. Since it comprises the feature that best map the input MTS data to the output labels, visualizing the 2D matrix as an image could offer hints or visual clues about the raw MTS data. In a one-channel image, the color intensity of each pixel is proportional to the corresponding entry's value in the associated matrix. One can think of the formed image as the visual representation of the input MTS signals. The 2D image obtained can be expressed as follows:

$$h^{L1+1} = [h_1^{L_1}, h_2^{L_1}, ..., h_K^{L_1}]^T \tag{7}$$

where K is the total number of 1D feature maps at layer L_1. The dimension of the generated 2D image is $K \times L'$. Assuming *valid* convolution (i.e., $P = 0$) is performed, then L' is computed as follows, assuming s and L_W are constants:

$$L' = \frac{L_{in} - L_W + \sum_{i=1}^{L_1 - 1} s^i (1 - L_W)}{s^{L_1}} \tag{8}$$

4.3 Visual feature extraction module

Unlike the visual representation learning component of the network, the input of the visual feature extraction module is an image (i.e., a 2D matrix) that corresponds to the visual representation of the raw MTS data. In this module, the one-channel image obtained from the previous module is visually recognized, and visual features and patterns are learned via 2D convolution operations. This component of the network is analogous to the first module; except that the 1D convolutions are replaced by 2D convolutions. The 2D convolution operations enable the network to capture the interactions and co-movements between the different time-series signals. We also highlight that each layer in this module creates abstractions based on the information propagated through previous layers; for example, the first layers usually extract low-level information while the deeper layers find more complex patterns. This module results in a set of 2D feature maps, each of which is formed by applying the 2D convolution operation on different receptive fields of the input 2D matrix. Note that different 2D kernels extract and detect different structures and patterns of the input visual data representation. In this way, different local correlations at different locations of the input image can be learned.

4.4 Classification module

This module is represented by a multi-layer perceptron (ML) network that takes in a flattened output from the previous module as input and maps it to a scalar. Flattening concatenates the 2D feature maps to form a flat structure that can be fed into an MLP network. The MLP network is a network in which the neurons from each layer are connected to all neurons in the next and previous layers. In the context of binary classification, the output layer of the MLP network consists of a single output

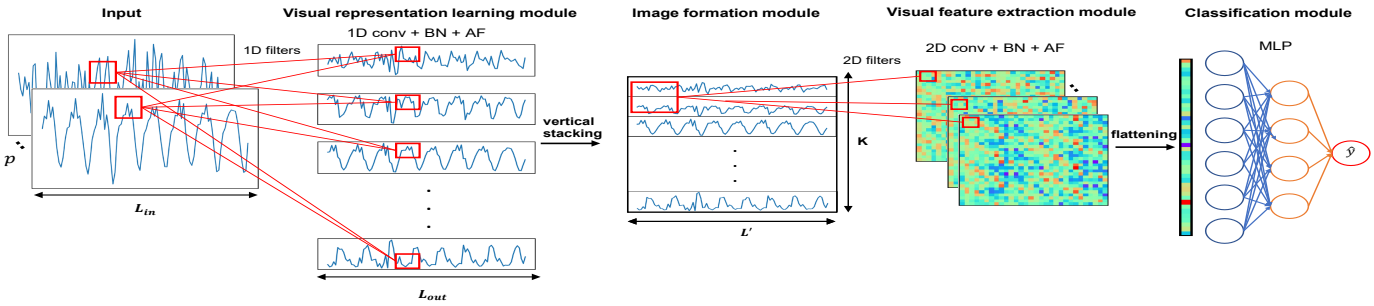

Fig. 6. *A sample of the proposed network configuration. Abbreviations: BN - batch normalization and AF - activation function.*

neuron $\hat{y}$ that indicates the class score (i.e., probability of a faulty event $P(y = 1)$) of the input MTS signals which can be described as the weighted sum of the hidden values from the last hidden layer, followed by a sigmoid function. Note that although we consider a binary classification problem in this work, the network can be extended for multi-class classification problems. For multi-class classification tasks, the number of output neurons of the network is equal to the number of classes and the output activation function is softmax function.

For classification tasks, the loss function represents the divergence of the predicted class score $\hat{y}$ (i.e., probability) from the actual class label y. Since this work deals with classification problems that have two output classes, a binary cross-entropy loss is used to measure the network performance and it can be expressed as:

$$L(W, b) = -\frac{1}{N_s} \sum_{i=1}^{N_s} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \quad (9)$$

where N_s represents the total number of input MTS signals, y_i is the true label for the i^{th} input MTS signal, and $\hat{y}_i$ is the predicted class probability. Next, the proposed network is trained by minimizing the loss function using an optimization algorithm. The optimization algorithm and its arguments are hyper-parameters to be specified before the training process.

5. INDUSTRIAL CASE STUDY

In this section, a case study of an industrial direct current electric arc furnace (DC EAF), a crucial part of the steelmaking process, is presented to further investigate the effectiveness of the proposed visual analytics framework for fault detection.

5.1 Process Description

DC EAF, as illustrated in Fig. 7, is widely used for steelmaking. Typical mining and metallurgical processes consist of three primary stages; the first is the ore extraction stage in which the ore is extracted from open pit mines using trucks and hydraulic shovels; the second is the ore preparation and feeding stage where the extracted ore undergoes several upstream processing (e.g., screening, crushing, drying, calcining, etc.) to provide a fine particulate DC EAF feed that is dried, reduced, and heated; the final stage is the smelting stage, whose role is to smelt and refine the processed ore into base metals using the DC

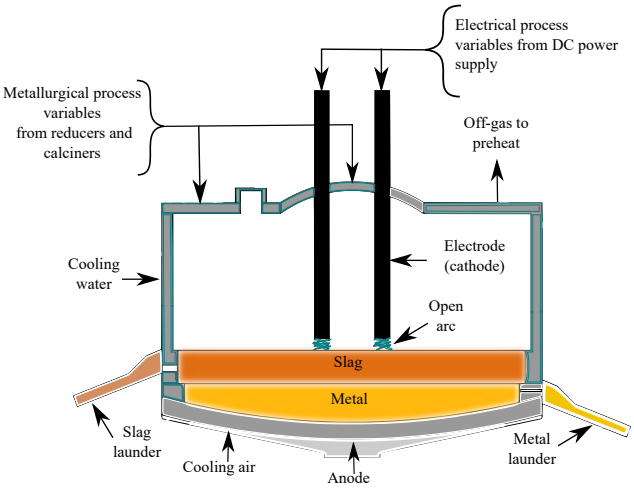

Fig. 7. *An illustration of a direct current electric arc furnace (Rippon et al. (2021)).*

EAF unit. More details about the process and the data pre-processing can be found in (Rippon et al. (2021)).

The overall DC EAF process can be summarized as follows: pulverized, processed ore is fed into the twin electrode DC EAF unit, a refractory-lined vessel with two tapping launders, for smelting under high-temperature conditions. In DC EAF, the electrical energy is converted into thermal energy mainly by the arcs that span from the top graphite electrode tips to the surface of the molten scrap metal (i.e., slag). The feed enters the furnace from multiple ports along the roof, whereas the slag and alloy are tapped intermittently from the launders.

To maximize production efficiency, decrease the variability in the final product, and increase profitability, the DC EAF operation needs to be stable. An unexpected loss of the plasma arc is a recurring and unresolved fault that significantly impacts the production rate and the electrical efficiency of the furnace. Electrical disturbances from the DC power supply, feed disturbances from the upstream metallurgical processes, and the operation of the EAF are three main probable causes of an arc loss.

5.2 Data description

The DC EAF process is a multivariate system with strong non-linearity and correlations between process variables. In this case study, an entire year of operation data is

collected from multiple upstream units, the DC EAF unit, and the power supply system. The sampling frequency of the measured variables is 3 seconds. The dataset is highly imbalanced with $> 99\%$ of the instances belonging to the no arc loss class (i.e., smooth operation, $Y = 0$) while only $< 1\%$ of the samples are labeled as ac loss (i.e., faulty operation, $Y = 1$). To address the class imbalance challenge, the majority class is randomly under-sampled. Hence, the final data consists of 3052 MTS signals with a balanced class ratio of 50:50. Each sample corresponds to 55 consecutive minutes worth of 96 different process variable measurements taken either during a smooth or faulty operating period. The objective is to develop a soft sensor to predict if a given MTS signal (i.e., 96 1D signals) of size 1100 (=55 operating minutes) belongs to a smooth operation or a faulty operation (i.e., arc loss event).

6. EXPERIMENTAL RESULTS & DISCUSSION

The experiments in this work mainly consist of training, validating, and testing five different models to predict the arc loss events in the DC EAF unit. The five models considered for this work are the following: i) 1D-CNN, ii) 2D-CNN on raw features (i.e., actual process measurements), iii) RP followed by 2D-CNN, iv) GAF followed by 2D-CNN, and v) our proposed model. The input data for the 1D-CNN model is a 96-channel 1D vector in which each channel corresponds to a different process variable. Next, the 96 1D signals are fused into a 2D matrix and used as an input for the 2D-CNN model. Our experimental set-up has two key objectives: i) investigating and validating the benefits of visual representation of MTS signals in solving fault detection tasks (i.e., MTS classification tasks) and ii) comparing the performance of the proposed method with respect to other state-of-art signal imaging tools (e.g., GAF and RP) on multivariate industrial operating data.

A hold-out strategy is employed for model evaluation which implies that the entire dataset is taken and split, chronologically rather than randomly, into three subsets: i) 70% training set (50:50), ii) 15% validation set (50:50), and iii) 15% testing set (50:50). Next, a random search over a manually predefined search space is performed to find a well-performing model configuration for each model. The number of trials used in the random search is manually chosen depending on the size of the search space, the size of the data, and the available computational power. Finally, models with the hyper-parameters configuration that achieved the best performance on the validation set during the random search are tested on the testing set, and the results presented in this section are reported on the final testing set.

The experimental results are summarized in Table 1. Five different evaluation metrics are used to evaluate each model's performance. Accuracy is simply defined as the sum of true predictions over the total number of testing samples. Moreover, the percentage of true predicted alarms over total alarms is defined as precision while TPR (i.e. recall) is defined as the number of times arc loss is correctly predicted divided by the total number of arc loss events. Next, FPR is defined as the number of false alarms per the total number of no arc loss events, and the F_1 score represents the harmonic mean of precision and recall. The most precise experimental configuration is with a 2D-CNN on raw features whereas our proposed approach has the highest score in four out of five key performance metrics.

Although 1D-CNN has a simple configuration and low-cost implementation since it consists of a series of 1D convolution operations, it fails to sufficiently capture the various local correlations among different process variables; as supported by the results. Moreover, directly re-arranging MTS signals into a 2D matrix of size (1100) as a 2D-CNN input has a major drawback which is that the various local correlations between process variables that are not within the same local receptive field are partially captured by the traditional CNN (i.e., the model performance is a strong function of the order at which the variables are listed in the fused 2D matrix). For imaging tools (e.g., RP and GAF), the computational complexity is directly proportional to a power law of the input size L_{in}. Hence, the raw MTS signals, of size 1100, are reduced to a fixed length of 110 using Piece-wise Aggregate Approximation (PAA) smoothing. Then, GAF and RP images are built on the smooth MTS signals (of size 110). The significant reduction in the signal size results in a loss of temporal information in the input MTS data; thus, RP and GAF are not able to capture the auto-correlations in the raw MTS data sufficiently. This explains the poor performance of the GAF-CNN and RP-CNN models. Next, our approach exploits the best characteristics of both 1D-CNN and 2D-CNN. It is capable of taking 1D signals of any size without the need to use PAA smoothing due to the simple configuration and low computational complexity of 1D-CNN. In addition, the 2D-CNN component helps the network to capture and recognize the temporal and local correlations in the input MTS signals.

For process monitoring applications, the adoption of ML models can't be assessed solely on their predictive power, they have to be able to provide a reasonable explanation for their decisions/ predictions. Process experts should be enabled to verify and examine the relatedness of the model's prediction to the actual process. We believe that the visual data representation could help lift the veil of deep learning algorithms (i.e., the so-called "black box" problem) by providing visual hints about the process conditions and incorporating process knowledge in the analysis process. Note that imaging tools like RP and GAF produce a 1-channel image for each univariate time-series signal; therefore, the visual representation of MTS data is a multi-channel image in which the number of channels is equivalent to the number of variables (i.e., the number of univariate time-series). Since normal computer monitors can't render images with more than three channels, one has to search for the set of three channels that best discriminate between the normal and faulty MTS signals which could be challenging for high dimensional systems. Fig. 8 shows only 3 channels of the RP and GAF images, of size 110×110, as RGB images constructed from the DC-EAF data during two different operating modes (I and II). RP and GAF adapt differently to different types of signals; therefore, one can visually differentiate between images corresponding to normal signals ($Y = 0$) and faulty signals ($Y = 1$) as shown in Fig. 8. Note that for multi-modal processes, RP and GAF will also produce different visual representations for signals taken from different

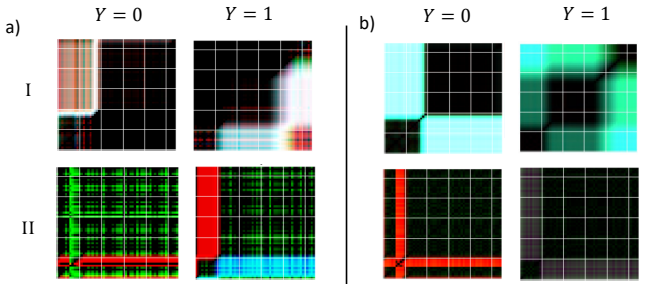

Fig. 8. *An illustration of: a) GAF encoding examples and b) RP images as RGB images obtained from DC EAF data for smooth signals $Y = 0$ and faulty signals $Y = 1$ at two different operating modes (I and II).*

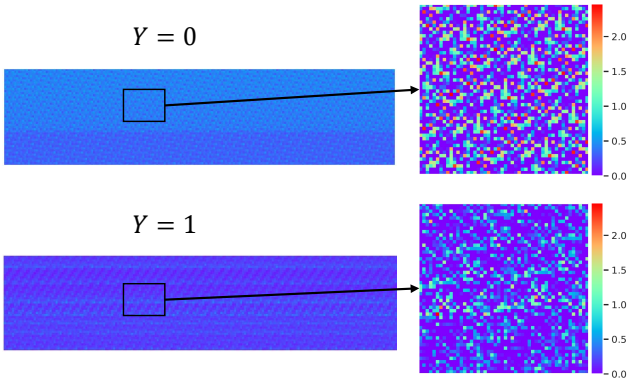

Fig. 9. *An illustration of the visual representation of DC EAF data obtained using the proposed approach for smooth signals ($Y = 0$) and faulty signals ($Y = 1$).*

operating modes. Moreover, Fig. 9 illustrates the visual representation extracted from the image formation module of the proposed network which is a one-channel image of size 1100×256 ($L' = L_{in} = 1100$ and $K = 256$). Since the size of the visual representation obtained in the image formation module is larger than GAF and RP images, the overall texture and typology are not as clear as in RP and GAF images. Finally, since our proposed approach yielded competitive results in detecting arc loss events in the DC EAF unit, it means that the visual analytics paradigm is capable of producing images from MTS signals that contain information needed to discriminate between healthy and faulty signals.

Table 1. Results summary

	1D-CNN	2D-CNN	GAF	RP	proposed
Accuracy	69.87%	71.18%	63.97%	65.72%	**75.76%**
Precision	62.28%	**72.00%**	66.15%	63.89%	71.97%
TPR	73.20%	70.13%	79.18%	70.93%	**79.63%**
FPR	32.58%	27.75%	57.67%	39.39%	**27.69%**
F_1	67.30%	71.05%	72.08%	67.22%	**75.60%**

7. CONCLUSION AND PROSPECTS

In this paper, we evaluated the application of two state-of-the-art time-series imaging approaches on a large industrial dataset in a supervised fault detection set-up. In addition, we introduced an end-to-end visual analytics workflow using 1D and 2D convolution operations. Existing and new methods are tested on a real industrial dataset to explore how MTS data can be encoded into pixelated images to reveal normal versus abnormal process operations. In other words, the problem of fault detection is configured as a visual analytics problem. Visual analytics is a promising research area that can offer significant benefits to the process industry. In future work, the application of visual analytics in data pre-processing and cleaning will be investigated.

ACKNOWLEDGEMENTS

We gratefully acknowledge the financial support from BBA Engineering Consultants and the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

Cook, K. and Thomas, J. (2005). *Illuminating the Path: The Research and Development Agenda for Visual Analytics*, volume 54.

Eckmann, J.P., Kamphorst, S.O., and Ruelle, D. (1987). Recurrence plots of dynamical systems. *Europhysics Letters (EPL)*, 4(9), 973–977.

Garcia, G.R., Michau, G., Ducoffe, M., Gupta, J.S., and Fink, O. (2021). Temporal signals to images: Monitoring the condition of industrial assets with deep learning image processing algorithms. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 1748006X2199444.

Hatami, N., Gavet, Y., and Debayle, J. (2017). Classification of time-series images using deep convolutional neural networks.

Hoang, D.T. and Kang, H.J. (2019). Rolling element bearing fault diagnosis using convolutional neural network and vibration image. *Cognitive Systems Research*, 53, 42–50. Advanced Intelligent Computing.

Jiao, J., Zhao, M., Lin, J., and Ding, C. (2019). Deep coupled dense convolutional network with complementary data for intelligent fault diagnosis. *IEEE Transactions on Industrial Electronics*, 66(12), 9858–9867.

Reis, M.S. and Gins, G. (2017). Industrial process monitoring in the big data/industry 4.0 era: from detection, to diagnosis, to prognosis. *Processes*, 5(3).

Rippon, L., Yousef, I., Hosseini, B., Bouchoucha, A., Beaulieu, J., Prévost, C., Ruel, M., Shah, S., and Gopaluni, R. (2021). Representation learning and predictive classification: Application with an electric arc furnace. *Computers & Chemical Engineering*, 150, 107304.

Shang, C. and You, F. (2019). Data analytics and machine learning for smart process manufacturing: Recent advances and perspectives in the big data era. *Engineering*, 5(6), 1010–1016.

Souza, V.M., Silva, D.F., and Batista, G.E. (2014). Extracting texture features for time series classification. In *2014 22nd International Conference on Pattern Recognition*, 1425–1430.

Springenberg, J.T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2015). Striving for simplicity: The all convolutional net.

Wang, Z. and Oates, T. (2015). Imaging time-series to improve classification and imputation.

Wu, H. and Zhao, J. (2018). Deep convolutional neural network model based chemical process fault diagnosis. *Computers Chemical Engineering*, 115, 185–197.

Xia, M., Li, T., Xu, L., Liu, L., and de Silva, C.W. (2018). Fault diagnosis for rotating machinery using multiple sensors and convolutional neural networks. *IEEE/ASME Transactions on Mechatronics*, 23(1), 101–110.

Yuan, X., Qi, S., Shardt, Y.A., Wang, Y., Yang, C., and Gui, W. (2020). Soft sensor model for dynamic processes based on multichannel convolutional neural network. *Chemometrics and Intelligent Laboratory Systems*, 203, 104050.

Zheng, Y., Liu, Q., Chen, E., Ge, Y., and Zhao, J.L. (2014). Time series classification using multi-channels deep convolutional neural networks. In F. Li, G. Li, S.w. Hwang, B. Yao, and Z. Zhang (eds.), *Web-Age Information Management*, 298–310. Springer International Publishing, Cham.