IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 16, 2020, accepted September 29, 2020, date of publication October 7, 2020, date of current version October 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3029323

ORANGE: Outcome-Oriented Predictive Process Monitoring Based on Image Encoding and CNNs

VINCENZO PASQUADIBISCEGLIE[1], ANNALISA APPICE[1,3], GIOVANNA CASTELLANO[1,3], (Member, IEEE), DONATO MALERBA[1,3], (Member, IEEE), AND GIUSEPPE MODUGNO[2]

[1]Department of Computer Science, University of Bari Aldo Moro, 70125 Bari, Italy
[2]MTM Project srl, 70043 Monopoli, Italy
[3]Consorzio Interuniversitario Nazionale per l'Informatica-CINI, 70121 Bari, Italy

Corresponding author: Vincenzo Pasquadibisceglie (vincenzopasquadibisceglie@uniba.it)

The work of Vincenzo Pasquadibisceglie was supported by the Programma Operativo Nazionale Ricerca e Innovazione (PON-RI)-Big Data Analytics for Process Improvement in Organizational Development-Codice Unico di Progetto (CUP) H94F180002600062014-2020. This work was supported in part by the Programma Operativo Regionale (POR) Puglia Fondo Europeo di Sviluppo Regionale-Fondo Sociale Europeo (FESR-FSE)-Bando Innolabs from 2014 to 2020, and in part by the Research Project Knowledge Community for Efficient Training (KOMETA) through Virtual Technologies funded by the Regione Puglia.

ABSTRACT The outcome-oriented predictive process monitoring is a family of predictive process mining techniques that have witnessed rapid development and increasing adoption in the past few years. Boosted by the recent successful applications of deep learning in predictive process mining, we propose ORANGE, a novel deep learning method for learning outcome-oriented predictive process models. The main innovation of this study is that we adopt an imagery representation of the ongoing traces, which delineates potential data patterns that arise at neighbour pixels. Leveraging a collection of images representing ongoing traces, we train a Convolutional Neural Network (CNN) to predict the outcome of an ongoing trace. The empirical study shows the feasibility of the proposed method by investigating its accuracy on different benchmark outcome prediction problems in comparison to state-of-art competitor methods. In addition, we show how ORANGE can be integrated as an Intelligent Assistant into a CVM realized by MTM Project srl company to support sales agents in their negotiations. This case study shows that ORANGE can be effectively used to smartly monitor the outcome of ongoing negotiations by early highlighting negotiations that are candidate to be completed successfully.

INDEX TERMS Predictive process analytics, outcome prediction, computer vision, convolutional neural networks, spatial data modeling.

I. INTRODUCTION

Nowadays predictive process mining is playing a fundamental role in the business scenario. In particular, it is emerging as an effective means to monitor the execution of any business running process by predicting at run-time the outcome of ongoing traces of a process given their uncompleted executions [1]–[6]. For instance, knowing in advance the outcome of an ongoing trace (e.g. the placement of a purchase order by a potential customer) may foster the early implementation of mitigation strategies, in order to increase the customer satisfaction, as well as to promote efficiency and quality of the business process management.

In general, outcome-oriented predictive process monitoring methods refer to classifying each ongoing trace of a process according to a given set of possible categorical outcomes. In this way the problem of outcome process monitoring is formulated as a problem of early sequence classification. According to this formulation, given a trace of an ongoing process execution (which can be modeled as a sequence of events with data payloads [7]), the outcome-oriented predictive process seeks to predict as early a possible whether the outcome of the trace will fall into a positive class or a negative class. For example, in an order-to-cash process, the positive class may include purchase orders that are closed successfully, while the negative class may correspond to canceled or withdrawn orders. Another set of possible outcomes is that the products were delivered on time (positive class) or delivered late (negative class).

The associate editor coordinating the review of this manuscript and approving it for publication was Bohui Wang.

The prediction of the correct class is based on a classification model extracted from historical process execution logs (event logs) [8]. In particular, current methods for outcome-oriented predictive process mainly follow the standard pipeline for classification involving two phases: (i) a feature extraction phase where features are extracted from event traces so that each trace is abstracted as a vector of features representing event occurrences and (ii) a classifier construction phase, where feature vectors are given as input to a machine learning method (e.g. decision trees) to produce a classifier.

Recently, the traditional machine learning pipeline for classification has been overcome by the deep learning approach in many application fields [9]. In general, deep learning architectures are computational models that are composed of multiple processing layers capable to learn representations of data with multiple levels of abstraction. From this point of view, deep learning methods overcome conventional machine learning methods, because of their ability to detect optimal features in raw data through consecutive nonlinear transformations, with each transformation reaching a higher level of abstraction and complexity of the extracted features.

Among the several deep learning architectures, Convolutional Neural Networks (CNNs) deserve special attention as they have dramatically improved the state of the art in the field of computer vision. Inspired by the amazing results of deep learning in computer vision, a very recent trend is bringing together computer vision and process mining [10]. This seminal study has in fact assessed the viability of CNN methods designed for computer vision into predictive process mining approaches for next activity prediction.

Boosted by this recent application of computer vision approaches in process mining, in this paper we introduce a novel deep learning approach for deriving a classification model useful for outcome-oriented predictive process monitoring. The proposed method, called **ORANGE (Outcome pRediction bAsed oN imaGe Encoding)** adopts an imagery representation of ongoing traces, which is able to delineate potential data patterns that arise at neighbour pixels representing trace features. Leveraging a collection of images representing running traces, a CNN is trained to predict the outcome of each trace.

The novel contribution of this study is the achievement of a new important milestone in coupling computer vision to process mining. In particular, we prove that computer vision methods can aid in gaining accuracy not only in the next activity prediction, as it was initially investigated in [10], but also in problems of outcome-oriented predictive process monitoring. This result is achieved once a vector of trace features is extracted and arranged as pixel frames of an image and CNNs are trained to address the outcome monitoring process as a problem of image classification.

On the other hand, focusing the attention on the technique we adopt here to transform traces in images, we further advance the previous study in [10] improving the effectiveness of the images of traces produced. In fact, in this paper, we take advantage of the ability of capturing possible patterns of spatial continuity among data features as described in [11] and use these patterns to transform traces into effective images.

Specifically, we identify trace features that assume similar values on training samples and associate them with neighbour pixel frames. We point out that data continuity among trace features was neglected in the past study [10], where trace features were associated to pixel frames following the naive order according to the features were extracted from the trace events. On the other hand, discovering and accounting for the existence of phenomena of data continuity among trace features allow us to construct images of traces, which plausibly exhibit continuous grey-scale intensities at neighbour pixels instead of salt and pepper intensities. This contributes to construct images depicting specific spatial data arrangements on neighbour pixels (e.g. edges, shading changes, shapes). These arrangements are common in real imagery data and may be helpful for the use of filtering and pooling operations when training CNNs on images of traces.

As an additional contribution, we evaluate the effectiveness of the proposed method in various outcome-oriented process monitoring problems for traces collected in benchmark event logs. The empirical study presented in this work investigates the ability of **ORANGE**, to increase accuracy when compared to several competitors taken from the recent literature. In addition, we present a case study to show the viability of the proposed method as a means to increase the value of the data-driven Customer Value Management (CVM) called **CRM** and realized by an Italian software company – MTM Project srl.[1] Coupling **ORANGE** to **CRM**, we are able to engage inputs and customers' performance data to predict success outcomes on their negotiations. The final feeling is that a new generation CVM integrating outcome-oriented predictive process monitoring may optimize business processes and decisions throughout the entertainment ecosystem. It also boosts content programming to realize a better return on their content investment.

The paper is organized as follows. An overview of related works is provided in the next Section. The proposed method is described in Section III. Section IV describes the experimental setup and discusses the results on different real-world event logs. Section IV-D illustrates the effectiveness on the outcome-oriented predictive process monitoring performed within a data-driven Customer Value Management (CVM). Finally, Section V draws conclusions and outlines future work.

II. RELATED WORK

There is a plethora of works addressing the problem of predicting the outcome of an ongoing trace of a business process based on event logs. In [5] a systematic review of outcome-oriented predictive process monitoring methods is presented.

[1] https://www.mtmproject.com/

Previous approaches to this problem are largely based on simple symbolic sequence classification, meaning that they extract features from traces seen as sequences of event labels, and use these features to construct a classifier for run-time prediction [1]–[3], [6], [12]. However, these approaches often ignore the data payload associated to each event. In [1] the payload of the last executed event is taken into account, but the evolution of data throughout the execution traces is ignored.

Other approaches treat traces as complex symbolic sequences, that is, sequences of events each carrying a data payload. For example, in [13] the authors consider traces as sequences of activities each carrying a data payload consisting of attribute-value pairs. By starting from this assumption, the authors compare different feature encoding approaches, ranging from traditional ones, such as counting the occurrences of activities and data attributes in each trace, up to more complex encoding that take into account also the evolution of data by combining Hidden Markov Models with an index-based encoding specifying, for each position in the case, the event occurring in that position and the value of each data attribute in that position. In [14], unstructured (textual) information, contained in text messages exchanged during process executions, is also considered, together with control and data flow information. A crucial phase in these approaches is how to encode a complex symbolic sequence in terms of vectors of features representative of the data payload.

As concerns the construction of the classification model, existing predictive process monitoring methods have adopted different machine learning algorithms, the most popular choice being Decision Tree (DT) [15]–[17], Random Forest (RF) [13] and SVM [18]. Also the XGBoost classifier showed promising results when applied to business process data [19]. In general, RF and boosted trees have shown to outperform other methods in many predictive monitoring scenarios, including the outcome prediction task [5].

However the recent success of deep learning approaches in computed vision for tasks of image classification has recently attracted attention also in process mining. Pioneering studies on deep learning for predictive process mining have mainly investigated Long Short-Term Memory (LSTM) [20]–[23] architectures to accomplish various process predictive tasks (next activity, time of next activity and completion time). LSTMs are mainly adopted in predictive process mining due to their natural ability in delivering consistently high accuracy in the natural sequence modeling of a trace without imposing flattening the data from the different events and loosing information on the order. In [24], an adversarial training framework is formulated to perform the next activity prediction with LSTMs. The framework is formulated to avoid sub-optimal network configurations and architectures due to insufficient training data. It adapts Generative Adversarial Networks (GANs) to the realm of sequential temporal data, where both the generator and the discriminator are LSTMs. Finally, the authors of [25] adapt the LSTM-based neural architecture defined in [22] for problems of next activity prediction, in order to predict the outcome of each ongoing trace in an event log.

In spite of the prevalence of predictive process mining studies with LSTMs, there are a few investigations where other deep neural network architectures, such as 1D Convolutional Neural Networks (CNNs) [26], Recurrent Neural Networks (RNNs) [27], [28] or stacked autoencoders [29], are applied in the realm of process mining. All these approaches are formulated for the next activity prediction and apply deep neural networks to training data that are feature vectors of event traces rather than sequences of events. However, none of these approaches embeds spatial structures, hence they are not able to take full advantage of filtering and pooling operations that are applied when training such networks on spatially structured data such as images.

In [30], starting from the evidence that 2D CNNs – CNN architectures defined to process 2D data as images, are very accurate classifiers whenever applied to image data embedding a clear spatial structure [31]–[34], the authors firstly explore the use of a 2D CNN to perform next activity by processing imagery representations of non-visual sequential data coming from traces of event logs. However in [30] only simple features (i.e. rough activities and timestamps) are encoded in the imagery representation by resembling more a multi-variate time series format rather than an image format.

In [10], a more complex imagery scheme is presented as a means to take into account multi-perspective trace features and derive a richer representation of trace events. In particular, trace samples are firstly converted into feature vectors that are subsequently represented in the form of RGB-like images, where pixels are associated with color values and each pixel captures a feature of the trace. The RGB images are finally used to train a 2D CNN based on Inception block [35] for next-activity prediction. The results collected in [10] provide empirical evidence of the effectiveness of resorting to the imagery trace representation in process mining. Although this study confirms that computer vision can successfully be applied to process mining, the imaging technique used in [10] leaves out any mechanism to explore possible continuity (or similarity) across values measured on various features of the same trace, as these features are arranged as pixels of RGB images. In fact, in [10] the images of the traces are produced by processing the trace features in the naive order they have been computed and assigning them to consecutive pixel frames of an imagery grid (moving from the left to the right, from the top to the bottom of the image). So, they typically show a salt and pepper arrangement of pixels. This is a limitation from the computer vision point of view, where images are typically expected to show some form of continuity in the intensity values of neighbour pixels. In any case, the above studies clearly indicate how well the potential of deep learning can be transferred from computer vision to predictive process mining. The ORANGE approach presented in this paper advances further the state of the art in coupling computer vision and process mining. ORANGE extends the imagery representation of ongoing traces adopted in [10]

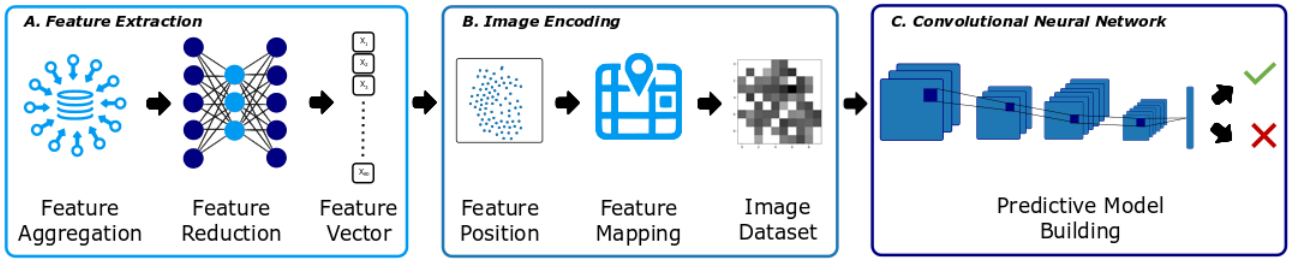


FIGURE 1. The ORANGE pipeline.

TABLE 1. A fragment of the example event log Production (see Section IV-A for details on this event log).

TraceId	EventId	Timestamp	Activity	Resource	...	Outcome class
σ_1	e_1	2012/01/29 23:24:00.000	Turning & Milling	Machine 4	...	
σ_1	e_2	2012/02/01 08:18:00.000	Laser Marking	Machine 7	...	
σ_1	e_3	2012/02/14 00:00:00.000	Lapping	Machine 1	...	
σ_1	e_4	2012/02/14 09:05:00.000	Lapping	Machine 1	...	
σ_1	e_5	2012/02/16 12:43:00.000	Final Inspection	Quality checker	...	
σ_1	e_6	2012/02/17 00:00:00.000	Packing	Packing	...	regular
σ_2	e_7	2012/01/03 00:00:00.000	Lapping	Machine 1	...	
σ_2	...	...	...	...	...	
σ_2	e_{15}	2012/03/30 10:35:00.000	Final Inspection	Quality Checker	...	deviant
σ_2	...	...	...	...		

so that data continuity patterns arise as neighbour pixels in the resulting images, thus modeling spatial dependencies between trace features and enabling an accurate prediction of process outcome classes.

III. PROPOSED APPROACH

Given a trace, outcome-oriented predictive process monitoring aims at predicting its class label expressing its outcome according to some business goal. To solve this problem, we propose a novel deep learning approach called ORANGE (Outcome pRediction bAsed oN imaGe Encoding).

We assume the availability of an event log containing a list of process traces and we need to create a labeled dataset from the given event log. Each process trace under consideration identifies the execution of a process instance and consists of a finite sequence of events $\sigma = \langle e_1, e_2, \ldots, e_n \rangle$ such that each event appears at most ones in the event log. Every event measures a vector of attributes (comprising, for example, the executed activity, the resource triggering the event, the timestamp at which the event has been started). Optional trace attributes, that remain the same throughout the whole trace, may be also associated with each trace. An example of a fragment of an event log is reported in Table 1. From each trace in the event log, we can derive several prefix traces σ^k with $1 \leq k \leq |\sigma|$. Hence, a trace is a complete process instance (started and ended), while a prefix trace is an instance in execution (ongoing trace). The labeled dataset includes all the prefixes of all the traces in the input event log, labeled with the outcome class (that is, the outcome achieved at the completion of the trace). Let us consider the complete trace σ_1 in Table 1 that comprises six events ($|\sigma_1| = 6$). It defines six prefix traces labeled with the class label "regular" as follows:

σ_1^1: (e1), regular
σ_1^2: (e1 e2), regular
σ_1^3: (e1 e2 e3), regular
σ_1^4: (e1 e2 e3 e4), regular
σ_1^5: (e1 e2 e3 e4 e5), regular
σ_1^6: (e1 e2 e3 e4 e5 e6), regular

Given a set of completed traces with their known class labels, ORANGE creates a classification model capable to distinguish between positive (regular) and negative (deviant) traces. Firstly, prefix traces in the event log are mapped onto a vector of features extracted over various perspectives (with one perspective for each attribute stored with trace events). An autoencoder is employed to reduce the dimensionality of the feature vector (see Section III-A). Then an encoding method is applied to transform the reduced feature vector into a 2D image representing the trace (see Section III-B). A 2D CNN architecture is finally trained on the collected labeled images representing ongoing traces, in order to learn a classification model capable to estimate the outcome class of an ongoing trace (see details in Section III-C). After training, the CNN takes as input the image encoding any ongoing trace and predicts the outcome class. The pipeline of the proposed approach is depicted in Figure 1 and detailed in the followings.

A. FEATURE EXTRACTION

Various trace encoding schemes – Last state, Aggregation and Index – have been synthesized in the process mining literature. A review is done in [5], where the authors compare the performances of these schemes with respect to the

accuracy of several pipelines of outcome-oriented process monitoring. The Last state scheme considers the last event in the prefix trace generating a feature for each event attribute. The Aggregation scheme considers all the events since the beginning of the prefix trace and applies aggregation operators to events' attributes. For numerical attributes of events, it computes the average and standard deviation. For categorical attributes of events, it commonly uses the one hot encoding and computes the frequency of each symbol. Finally, it adds the trace features to the feature vector "as is" without any loss of past information. The Index scheme also uses all possible information (including the order) in the prefix trace, by generating one feature per each event attribute and per each executed event. We note that the Last state schema looses the information on the oldest events in the prefix trace, while both the Aggregation and Index schemes exploit information from all the performed events. However, the Aggregation still exhibits information loss by neglecting the order of the events. On the other side, a drawback of the Index schema is that due to the fact that the length of the feature vector increases with each executed event, this encoding can only be used when all samples have the same length through padding.

So, by accounting for the considerations formulated above, as well as the overall results of the empirical validation described in [5], we use here the trace encoding schema based on the Aggregation, in order to represent all prefix-traces as fixed length feature vectors of features. In fact, the comparison of the performances of Last state, Aggregation and Index done in [5] with respect to predictive pipelines with Random Forest and XGBoost evaluated in various problems of outcome-oriented process monitoring concludes that the Aggregation achieves the highest accuracy in the majority of the datasets. However, our decision of computing the Aggregation scheme differs from the one taken in [25], where the authors use the Index schema with the LSTM-based pipeline they have defined for outcome-oriented process monitoring. In any case, preserving information on the order is mandatory with LSTMs, as these are deep learning architectures formulated to process "sequence" data. We note that in this paper we opt for a different deep learning architecture, as we train 2D CNNs in place of LSTMs. This leads to process images instead of sequences, reducing the need of data enhancing the knowledge hidden in the order, while strengthening the request of data depicting spatial data continuity to fuel effective convolution operations.

Additional considerations concern the fact that the size of the vector of features constructed with Aggregation could be too large when process event dictionaries are big. In such cases, there is a need to reduce the feature space, to the complexity on the subsequent steps under control. To this aim, we resort to an autoencoder [36] that is a non-linear generalization of Principal Component Analysis (PCA). Autoencoders are neural networks that are trained to reconstruct their input [37]. More precisely, an autoencoder is composed of two modules: an encoder, which learns a nonlinear mapping between the input data and a smaller hidden latent space, and a decoder, which learns to reconstruct the original input by using features of the latent space. Hence the main objective of an autoencoder is to compress the input into a lower-dimensional code and then reconstruct the output from this representation. The results of this compression is called the latent-space representation. To learn a proper code, the parameters (weights and biases) of the network are optimized in both encoding and reconstruction phase to minimize a reconstruction loss.

Using an autoencoder, we map the features of event traces to a latent feature space that is suitable for the task of outcome class prediction, since we retain non-linear relationships among data. A similar approach is investigated in [29] for business process event prediction, where stacked autoencoders are applied to extract features from the pre-processed business process log data.

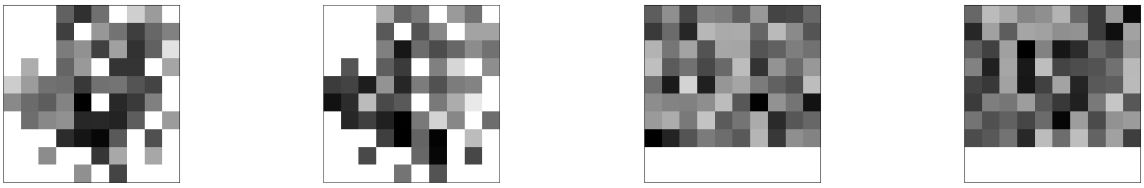
B. IMAGE ENCODING

Once the prefix traces have been transformed into the vectors of the above-described features, these vectors are transformed into images according to a 2D grid of $N \times N$ pixel frames. The core of this image encoding is the decision of how features of traces are associated to pixel frames in a way to delineate potential patterns of data continuity, which arise at neighbour pixels.

In an attempt to capture possible spatial relationships between features, the one-to-one association between features and pixel frames is done according to the theory introduced in [11]. Let us consider the training set θ composed of n samples (all prefix traces extracted for the training stage) spanned over the vector of m features (extracted with the feature extraction). θ is a data matrix with size $n \times m$. Let us define the set ψ – the data matrix with size $m \times n$ – which is constructed by transposing θ so that samples are put on columns and features on rows. The t-SNE [38] dimensionality reduction technique[2] is applied to reduce columns of ψ and project all features on two columns (determined as a non linear combination of the training samples where the features have been measured). The rows of ψ_{t-SNE} represent the location of the features in the Cartesian plane defined by these two columns. Subsequently, the convex hull algorithm is used to find the smallest rectangle containing all the point and a rotation is performed to framed the grid in a horizontal or vertical form. Finally, the Cartesian coordinates are converted to pixels frames by segmenting the rotate convex hull into $N \times N$ equally-sized rectangular pixel frames.

Whenever more than one features are associated to the same pixel frame, the feature with the highest mutual info is selected to be assigned to the pixel frame, while the remaining colliding features are filtered-out. This procedure to manage of collisions is different from the one described in [11], where colliding features were simply averaged. However, this solution may lead to summing-up values that measure

[2]In principle any non-linear dimensionality reduction techniques can be used to this aim.



(a) Regular trace (ORANGE) (b) Deviant trace (ORANGE) (c) Regular trace (naive method) (d) Deviant trace (naive method)

FIGURE 2. The image representation of two completed traces extracted from Production (see Section IV-A for details on this event log).

completely different features. On the other hand, the mutual-info approach we propose to apply can elegantly solve collisions by also achieving a feature selection goal.

Once the location of the features is established in the imagery grid according to the procedure described above, this will be the same for all the prefix traces (also the one used for testing). For each prefix trace, the value actually assigned to the pixel of its grey-level imagery representation is the value measured by the corresponding feature in the prefix trace. An example of the image encoding of both a regular prefix trace and a deviant prefix trace is shown in Figure 2. In particular, Figure 2a shows the image of a prefix trace labeled as regular, while Figure 2b shows the image of a prefix trace labeled as deviant. Both images are produced using the image encoding technique described above. On the other hand, Figure 2c and 2d show the images of the same samples, but they are produced using the naive encoding technique. The naive encoding assigns values of consecutive sample features to consecutive pixel frames by proceeding from the left to the right and from the top to the bottom of the 2D grid. We note that the images of the normal sample feel different from the images of the deviant sample independently of the image encoding technique used. However, a progressive gradation of grey continuity is evident on the neighbour pixels of the images built with the encoding technique used ORANGE, while a salt and pepper grey distribution emerges in the images built with the naive encoding.

C. CONVOLUTIONAL NEURAL NETWORKS

The images extracted from the prefix traces of a complete event log are used as training set to learn a classification model for outcome process prediction. To learn the model we use a 2D deep Convolutional Neural Network (CNN) architecture. CNNs extend simple fully connected feed-forward neural networks by introducing convolution and pooling operators, that are applied in several layers stacked on top of each other [39].

A convolutional layer applies a set of local filters that are replicated along the whole input to process small parts of the input and extract local features. Each convolutional layer receives the input (image) and convolves it by applying a set of filters. Each filter is a squared matrix of size s containing weights w_{j0} that acts as a local kernel on the input space, i.e. it can be modeled as a node that receives input only from a limited portion of the whole input (the receptive field of the node) and is thus suited to exploit local spatial correlations hidden in the input. Each node j applies the convolution operator over the input image, by computing the inner product of the filter at every location in the image, and outputs the result as a feature map $h_j(x, y)$. A non-linear function $f()$ is then applied to each feature map providing activation values:

$$a_j^{(l)} = f(h_j^{(l)}(x, y)).$$

For the first convolutional layer the values of the feature maps $h_j^{(1)}(x, y)$ are obtained by convolving the input map $I(x, y) = h^{(0)}(x, y)$ with the respective kernel $w_{j0}^{(1)}$ followed by the non-linear activation function:

$$h_j^{(1)}(x, y) = f\left(\sum_{(u,v)\in U} w_{j0}^{(1)} h^{(0)}(x + u, y + v) + b_j^{(1)} \right),$$

where $U = \{(u, v) \in \mathbb{N}^2 | 0 \leq u \leq s, 0 \leq v \leq s\}$ and $\mathbb{N}$ is the set of whole numbers. Theoretically, any non-linear function can be used as activation function, provided that it is continuous and differentiable, as required by the back-propagation learning algorithm. The most common choices are tanh, logistic, softmax and relu. In our case we used the Relu (Rectifier Linear Unit) function $f(x) = max(0, x)$ that is widely used in CNNs because it improves convergence of the network training with respect to sigmoid units [40] and because it limits the gradient vanishing downside as its by-product is usually one once x is positive.

A pooling layer [41] is added on top of each convolutional layer to achieve spatial invariance and to reduce the dimensionality of the feature maps, preserving important information and discarding irrelevant details. Indeed, a pooling layer generates a lower resolution version of the convolutional layer output. This adds translation invariance and tolerance to minor differences of patterns in the input. The pooling operation is typically a sum, average, maximum or even a combination of various methods. In this work we use the max pooling (taking the maximum value of filter activation from different positions within a specified window) since it is the best performing operator according to the literature [41]. Activation values of the max-pooling layer are divided into M bands. Each band receives input from r neighbouring bands of the convolution layer output, being r the pooling size,

to generate J values representing the maximum activations received from the J convolution filters within these r bands. By doing this maximization operation every n bands (where n is the sub-sampling factor) the max-pooling layer produces an output that is a lower resolution version of the convolution layer output. The m-th band of the max-pooling layer produces a vector of J activation values:

$$\mathbf{p}_m = [p_{m,1}, p_{m,2}, \cdots, p_{m,J}]^T,$$

where each activation is computed as:

$$p_{m,j} = \max_{k=1}^{r}(h_{m\times n+k,j}).$$

As a result a smaller number of bands are obtained. They provide lower resolution features containing more useful information that can be further processed by higher layers of the network. Higher layers use more broad filters that work on lower resolution inputs to process more complex parts of the input. Top fully connected layers finally combine inputs from all positions to do the classification of the overall inputs. This hierarchical organization enables a CNN to model local structures in the input once proper filter parameters are optimized via supervised learning algorithms.

D. IMPLEMENTATION DETAILS

ORANGE, whose code is publicly available on the GitHub[3] repository, is implemented in Python 3.6.9 – 64 bit version – using Keras 2.3.1[4] library that is a high-level neural network API using TensorFlow 1.15.0[5] as the back-end. The code of the feature extraction stage is that provided by [5] on https://github.com/irhete/predictive-monitoring-benchmark.

A min-max normalization is applied to the data computed by the feature extraction stage before applying the autoencoder for dimensionality reduction.

In the implemented autoencoder, the encoder starts with a layer sized equal to the the number of features extracted by feature extraction stage and compresses the features through a sequence of intermediate encoder layers e with size $size(e) = size(e - 40)$ until the bottleneck layer with size equal to 80. The decoder starts in the bottleneck layer (the output layer of the encoder) and maps the bottleneck signals back to the input space through symmetric decoder layers d with size $size(d) = size(d-1)+40$, respectively. For each intermediate layer of the autoencoder, we apply a tanh activation function. We adopt the Adam optimization algorithm with *mean squared error* as loss function.

The vectors of features extracted from the latent-space of the autoencoder are subsequently transformed into 10×10 grey-level images (as described in section III-B), which are used to train the 2D CNN.

The adopted CNN architecture consists of a series of pairs of *Convolutional* and *MaxPooling* layers. An increasing number of filters (32, 64) is applied to each convolutional layer.

[3]https://github.com/vinspdb/ORANGE

[4]https://keras.io/

[5]https://www.tensorflow.org/

TABLE 2. Configuration of neural network hyperparameters.

Parameters	Value Range
Learning Rate	$[0.00001, 0.0001]$
Number of layers	$[1, 2]$
Batch size	$[2^6, 2^{10}]$

The number of layers applied during the training phase is an hyper-parameter that is automatically selected with tree-structured Parzen estimator (TPE) [42] during the training stage. The hyper-parameters optimization phase has been conducted by considering the 20% of the training set as a validation set. Table 2 reports the hyper-parameters optimized during this stage and the corresponding range of possible values explored with TPE.

To speed-up the training of the CNN, we introduce Batch Normalization layers that normalize the output of each convolution layer before entering in the subsequent max-pooling layer. The output computed from these layers is passed to a *GlobalMaxPooling* layer that is given as input to the *Sigmoid* function to estimate the outcome class. The loss function adopted to measure the error between the expected label and the probability predicted by the neural network is the *Binary Cross Entropy*. The training of the neural network is accomplished through the Back Propagation learning algorithm.

The Backpropagation training is applied with early stopping to avoid overfitting. Specifically, the training phase is stopped when there is no improvement of the loss on the validation set for 20 consecutive epochs. Finally, Adam (Adaptive moment estimation) optimizer [43] is adopted to minimize the loss function. We fix to 200 the number of epochs.

IV. EXPERIMENTAL RESULTS

A. EVENT LOGS

The experiments are performed on seven outcome-prediction problems formulated from three real-life event logs. The public logs are accessible from the 4TU Centre for Research Data.[6] These outcome prediction problems have been selected among those experimented in [5], where the Aggregation schema (that we have also considered in this paper for the feature extraction phase) achieved the highest performance.

1) SEPSIS

This log [44] records trajectories of patients with symptoms of the life-threatening sepsis condition in a Dutch hospital. Each trace collects the sequence of events since the patient's registration in the emergency room until the patient's discharge from the hospital. Among others, laboratory tests together with their results are recorded as events. In any case, the reason of the discharge is available in the data in an obfuscated format. The authors of [5] defined three different labeling procedures for this log: (i) sepsis_1 where a patient returns to the emergency room within 28 days from

[6]https://data.4tu.nl/

the discharge, (ii) sepsis_2 where a patient is (eventually) admitted to intensive care and (iii) sepsis_3 where a patient is discharged from the hospital on the basis of something other than Release A (that is, the most common release type).

2) BPIC2011

This log [45] was published into the Business Processing Intelligence Challenge (BPIC) in 2011. It contains data coming from a Dutch Academic Hospital. Each trace describes the medical history of a given patient so that the applied procedures and treatments are recorded as activities. As in [5], we consider four constraint conditions and define four binary outcome prediction tasks accordingly (i.e. bpic2011_1, bpic2011_2, bpic2011_3 and bpic2011_4). In each binary task, a trace is assigned to outcome equal to 1 if the constraint is violated, to 0 otherwise. Details on the used constraint formulation are reported in [5].

3) BPIC2012

This log [46] was published into the Business Processing Intelligence Challenge (BPIC) in 2012. It contains the execution history of a loan application process in a Dutch financial institution. Each trace in this log records the events related to a loan application. The authors of [5] identified a multi-class labeling for this log based on the final outcome of a trace, that is, whether the application is accepted, rejected or canceled. Based on this multi-class labeling, three separate binary outcome prediction tasks have been defined referred to in [5] as bpic2012_1, bpic2012_2 and bpic2012_3.

4) PRODUCTION

This log [47] contains data from a manufacturing process. Each trace records information about the activities, workers and/or machines involved in producing an item. The labeling (production) is based on whether or not the number of rejected work orders is larger than zero.

B. EXPERIMENTAL SET-UP

We have reproduced the experimental setting introduced in [5]. A temporal split is used to divide the event log into train and test traces. To this aim, the traces of a log are sorted by the the starting timestamp. The first 80% are selected for training the predictive model, while the remaining 20% are considered to evaluate the performance of the learned model on the unseen traces. We note that this allows us to simulate the real-life situation where prediction models are trained using historic data (started before a given data) and applied to ongoing traces. Whenever some events in the training traces overlap with the testing period, training traces with events that overlap with the testing period have been discarded.

Different metrics can be used to measure the accuracy of predictions. Rather than returning a hard prediction (a binary number) on the expected case outcome, the classifiers usually output a real-valued score, reflecting how likely it is that the sample will end in one way or the other. Accounting for the considerations reported in [5], good outcome classifier will give higher scores to ongoing traces that will end with a positive outcome, and lower values to those ending with a negative one. Therefore, similarly to [5], we use the area under the ROC curve (AUC) metric that expresses the probability that a given classifier will rank a positive case higher than a negative one. A major advantage of the AUC metric over the commonly used accuracy, e.g. F-score (the harmonic mean of precision and recall), is that the AUC remains unbiased even in case of a highly imbalanced distribution of class labels [48]. Furthermore, AUC is a threshold-independent measure, as it operates on the ranking of the scores rather than the binary class values. Still, relying on a single evaluation criterion may provide a biased viewpoint of the results; therefore, we also report the F-scores additionally to AUC.

Overall AUC are F-score are both obtained by first computing the scores separately for each prefix length and then by taking the weighted average of the obtained scores, where the weights are assigned according to the number of prefixes used for the calculation of a given score. This weighting assures that the overall metrics are influenced equally by each prefix in the testing set, instead of being biased towards longer prefixes (i.e., where many traces have already finished).

In order to measure the earliness of the predictions, we also monitor the accuracy of the predictive model separately for each prefix length [13]. In each step, the prediction model is applied to a subset of prefixes of exactly the given length. The improvement of prediction accuracy as the prefix length increases should provide an implicit notion of earliness.

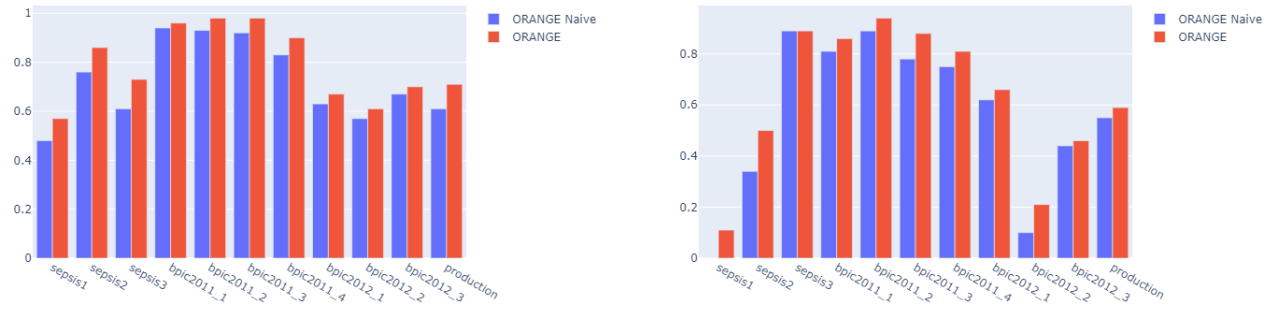
C. RESULTS AND DISCUSSION

We start the validation by investigating the effectiveness of the image encoding technique that we adopt in this study to transform trace data into images. To this purpose, we compare the performance of **ORANGE** to that of its variant **ORANGE–naive** that is defined replacing the image encoding technique used in **ORANGE** with that adopted in [10]. As the pipeline described in [10] is formulated for problems of next activity prediction, the validation is done using the 2D CNN architecture formulated in this paper for the outcome-oriented process monitoring. The overall AUC and F-score, of **ORANGE** and **ORANGE–naive** are compared in Figures 3a and 3b, respectively. This comparative analysis confirms that **ORANGE** systematically gains accuracy with respect to its variant **ORANGE–naive** by taking advantage of the better quality of the images produced. Specifically, this result provides the empirical evidence that **ORANGE** effectively benefits from its ability of accounting for phenomena of data continuity among features within the imagery encode done.

We proceed this validation by comparing the performance of **ORANGE** to that of the list of competitors – based on **SVM**, **LR**, **RF** and **XGB** – introduced in [5],[7] as well as the competitor – based on LSTM – described in [25].[8] **SVM**, **LR**, **RF** and

[7]The code of the competitors is available at https://github.com/irhete/predictive-monitoring-benchmark

[8]https://github.com/irhete/stability-predictive-monitoring



(a) AUC (b) F1-score

FIGURE 3. AUC and F-score of ORANGE and its variant ORANGE-naive on all the outcome prediction problems.

TABLE 3. AUC for both ORANGE and the competitors presented both in [5] (SVM, LR, RF, XGB) and in [25] (LSTM) on all the outcome prediction problems. The best results are in bold.

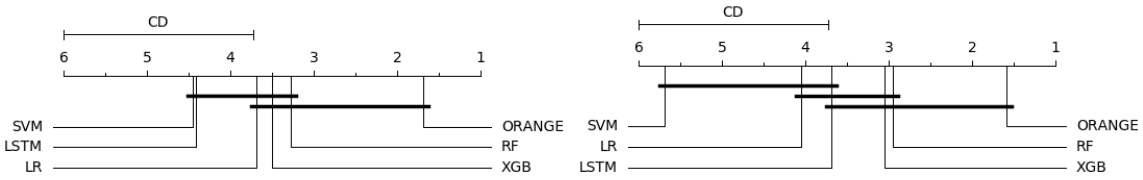
LOG	SVM	LR	RF	XGB	LSTM	ORANGE
sepsis1	0.49	**0.57**	0.41	0.33	0.52	**0.57**
sepsis2	0.82	**0.86**	0.79	0.85	0.73	**0.86**
sepsis3	0.72	**0.73**	0.66	0.72	0.61	**0.73**
bpic2011_1	0.87	0.92	0.94	0.94	0.93	**0.96**
bpic2011_2	0.95	0.94	**0.98**	**0.98**	0.91	**0.98**
bpic2011_3	0.96	0.96	**0.98**	**0.98**	0.93	**0.98**
bpic2011_4	0.87	0.87	0.89	0.86	0.89	**0.90**
bpic2012_1	0.63	0.65	0.69	**0.70**	0.62	0.67
bpic2012_2	0.55	0.59	**0.61**	0.57	0.60	**0.61**
bpic2012_3	**0.70**	0.69	**0.70**	0.69	**0.70**	**0.70**
production	0.66	0.67	0.63	0.70	0.67	**0.71**
avg	0.75±0.16	0.77±0.15	0.75±0.18	0.76±0.20	0.74±0.15	**0.79±0.15**

TABLE 4. F-Score for both ORANGE and the competitors presented both in [5] (SVM, LR, RF, XGB) and in [25] (LSTM) on all the outcome prediction problems. The best results are in bold.

LOG	SVM	LR	RF	XGB	LSTM	ORANGE
sepsis1	0.0	0.09	0.0	0.0	0.05	**0.11**
sepsis2	0.0	0.47	0.39	0.42	0.37	**0.50**
sepsis3	0.0	0.37	0.34	0.35	0.87	**0.89**
bpic2011_1	0.73	0.83	**0.86**	**0.86**	0.83	**0.86**
bpic2011_2	0.91	0.9	**0.95**	**0.95**	0.85	0.94
bpic2011_3	0.0	0.86	**0.94**	**0.94**	0.74	0.88
bpic2011_4	0.35	0.75	0.80	0.78	0.79	**0.81**
bpic2012_1	0.38	0.53	0.64	0.59	0.60	**0.66**
bpic2012_2	0.0	0.13	0.17	0.16	0.15	**0.21**
bpic2012_3	0.20	0.30	0.42	0.36	0.31	**0.46**
production	0.54	0.58	0.54	0.59	**0.62**	0.59
avg	0.28±0.33	0.53±0.29	0.55±0.32	0.55±0.32	0.56±0.31	**0.63±0.28**

XGB are run using the optimization described in [5] to select the hyperparameters of the machine learning algorithms. In addition, similarly to ORANGE, they use the Aggregation schema for the trace feature extraction. The decision of using the Aggregation schema with these competitors follows the conclusions drawn in the empirical study discussed [5]. On the other hand, LSTM is run with the architecture hyperparameters set as described in [25] and the trace feature extraction performed with the Index schema. In this case, the decision of coupling the LSTM architecture with the Index schema follows the formulation of the competitor reported in [25], as well as the consideration that, differently from the Aggregation schema, the Index schema is able to preserve the information on the order that the LSTM architecture is able to process.

The overall AUC and F-score of both ORANGE and competitors are reported in Tables 3 and 4, respectively. These results show that ORANGE commonly outperforms (or performs equally to) the competitors. The only exceptions concern the AUC of ORANGE on bpic2012_1 and the F-score of ORANGE on bpic2011_2 and bpic2011_3, which are lower than those of RF and XGB, while the F-score of ORANGE on Production is lower than that of LSTM. The interesting result is that, left-out ORANGE, there is no competitor that



(a) AUC (b) F1-score

FIGURE 4. Comparison of AUC and F-score of ORANGE, as well as the competitor approaches LR, RF, SVM, XGB with the Nemenyi test. Groups of approaches that are not significantly different (at $p \leq 0.05$) are connected.

systematically achieves the highest accuracy (AUC, as well as F-score) in all the problems. For example, considering AUC, LR is the best competitor on sepsis_1, sepsis_2 and sepsis_1, XGB is the best competitor on bpic2012_1 and production, RF is the best competitor on bpic2012_2, both XGB and RF are the best competitors on bpic2011_1, bpic2011_2 and bpic2011_3, while both RF and LSTM are the best competitors on bpic2011_4 and bpic2012_3. Similar conclusions can be drawn analyzing the F-score. So, there is no competitor that performs equally well on the various business processes. Differently, our deep learning proposal can maintain the same architecture design being equally effective in multiple business process scenarios.

To statistically test whether these considerations are statistically significant, we use the Friedman's test. This is a non-parametric test that is commonly used to compare multiple approaches over multiple datasets [49]. In particular, the Friedman test compares the average ranks of the approaches, so that the best performing approach gets the rank of 1, the second best gets rank 2. The null-hypothesis states that all the approaches are equivalent. Under this hypothesis, ranks of compared approaches should be equal. In this study, we perform this test by considering the average F-measure achieved by the compared approaches on each dataset and reject the null hypothesis with $p\text{-}value \leq 0.05$. As the null-hypothesis is rejected, that is, no approach is singled out, we use a post-hoc test—the Nemenyi test—for pairwise comparisons [49]. The results of this test, reported in Figure 4, confirm that ORANGE is ranked higher than the competitors. In addition, the resulting critical difference diagram, obtained using a 0.05 significance level, confirms that ORANGE is on average the best performing approach with respect to both AUC and F-score with RF as runner-up with respect to both AUC and F-score. This is coherent with the conclusions we have drawn before indicating ORANGE as the best-performing classifier in the addressed outcome prediction problems. For each problem, ORANGE is able to perform equally well to the best competitor that is not necessarily the same for all the considered problems.

At the completion of this study, we investigate the performance of ORANGE in terms of earliness. Figure 5 presents the AUC for the outcome prediction problems of sepsis (sepsis1, sepsis2 and sepsis3) evaluated across different prefix lengths. Each evaluation point includes prefix traces of exactly the given length. Thus the number of traces used for evaluation is monotonically decreasing when increasing prefix length. From Figure 5 it can be seen that ORANGE yields an AUC that is systematically greater than 0.5 (i.e., better than random) on each prefix length. On the other hand, there are a few competitors (e.g. LSTM in sepsis2 and sepsis3, SVM, RF and XGboost in sepsis1) whose performance may become worse than the random choice across some prefix lengths. Finally, in all the problems, ORANGE is the most accurate method on several prefix lengths. In general, it is systematically in the top-three ranked methods.

D. APPLICATION TO A CVM CASE STUDY

In this section we show the viability of the proposed method as a means to increase the value of a data-driven Customer Value Management (CVM). We present the application of ORANGE to CRM[9] that is a CVM realized by the Italian software company MTM Project srl to support negotiation and sales management. We processed data of 1388 events logged on 29 months during the execution of 140 traces of the negotiation and sales management process into CRM. Each event comprises information on the activity carried out, the sales agent triggering the activity and the timestamp. The labeling of the trace is positive (''accept'') if the sales negotiation is successfully completed, negative (''reject'') otherwise. Figure 6 shows the graphical representation of this log. This is extracted in the form of a direct follow graph using Directly Follows Graph plugin in ProM 6.9.[10]

This case study is intended to evaluate the viability of integrating ORANGE in CRM. In particular, the expectation of MTM Project srl is that ORANGE may be considered to process the historical event log stored by CRM and mine accurate outcome predictive knowledge from the logged events. This knowledge will allow the Intelligent Assistant of CRM to smartly assist the sales agent by suggesting the most promising ongoing negotiations (i.e. the negotiations that are candidate to be successfully completed). We note that, from the point of view of the sales agent, it is desirable to count on a predictive service that is as more accurate as possible in the early stages of the negotiation operations. This capability will

[9]https://www.mtmproject.com/cvm/
[10]http://www.promtools.org/doku.php

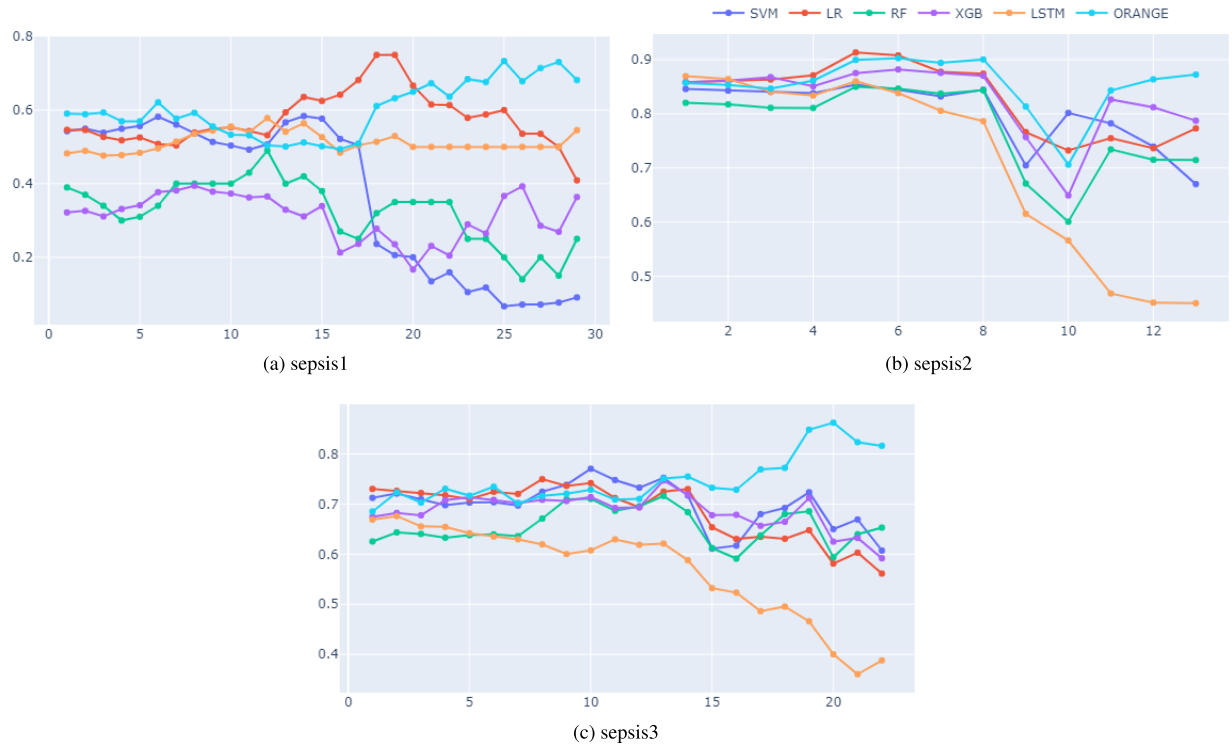


FIGURE 5. AUC (axis Y) on the outcome prediction problem of datasets sepsis1, sepsis2 and sepsis3 across different prefix lengths (axis X).

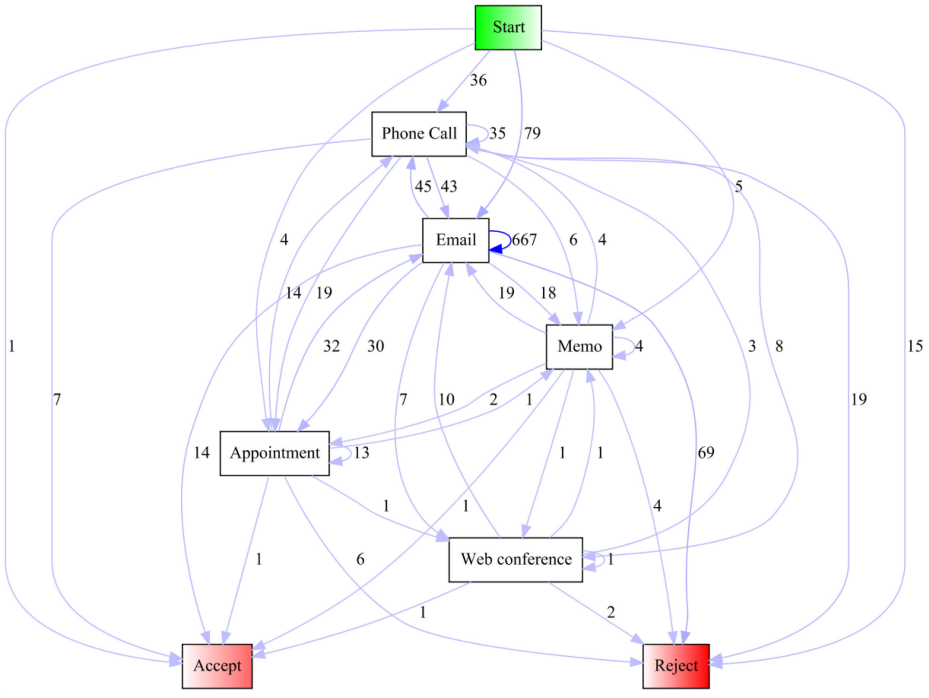


FIGURE 6. Direct Follow Graph of the event log collected with CVR.

actually support every sales agent of CRM in improving the workload organization by maximizing efficiency and profit.

As for the the benchmark outcome-oriented process monitoring problems analyzed in Section IV, the evaluation is done here splitting the event log of CRM into training traces (80%) and testing traces (20%). We train both ORANGE and the competitors on the training traces and evaluate the accuracy of the learned predictive models on the testing traces.

Table 5 reports the overall AUC and F-score of the methods compared in this case study. In addition, Figure 7

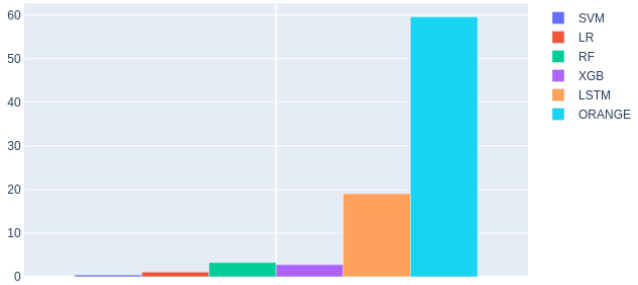


FIGURE 7. Computation time (in seconds) spent in completing the learning stage of ORANGE and the competitors presented both in [5] (SVM, LR, RF, XGB) and in [25] (LSTM) on the outcome prediction problem of CVR.

TABLE 5. AUC and F-score for both ORANGE and the competitors presented both in [5] (SVM, LR, RF, XGB) and in [25] (LSTM) on the outcome prediction problem of CVR. The best results are in bold.

Metric	SVM	LR	RF	XGB	LSTM	ORANGE
AUC	0.59	0.62	0.56	0.55	0.58	**0.66**
F-score	0.79	0.69	0.77	0.75	0.74	**0.80**

compares the methods in terms of computation time required to learn the outcome prediction models. The computation times are collected on a Linux machine with an Intel® Core i9-8950HK CPU @ 2.90GHz × 12 and 16GB RAM - NVidia GeForce GTX 1050 Ti. The compared times highlight that ORANGE spends more computation time learning the prediction model than the competitors. The higher computation complexity of ORANGE is mainly due to the image encoding step coupled to deep learning. In fact, we note that the training of LSTM which (similarly to ORANGE) is based on a deep learning architecture takes more time than SVM, LR, RF and XGB, which use traditional machine learning approaches. On the other hand, the learning stage of LSTM requires less computation time than ORANGE due to the lack of the image encoding step.

However, the higher training time of ORANGE is fully counterbalanced by the highest accuracy of the learned models which overcome all other competitors.

At the completion of this study, we evaluate the earliness of the predictions in the case study. Figure 8 shows the AUC computed along the different prefix lengths. These results confirm that, as desired, ORANGE outperforms all its competitors along the early stages of the test negotiations. This can aid sales agents in promptly identifying the most promising negotiations to be worth considering. We also note that ORANGE systematically outperforms competitors, except for the one based on SVM, also in the remaining stages of the negotiations. In any case, the SVM-based competitor performs more accurately than ORANGE in few cases, i.e. when the number of events already observed in the ongoing negotiation is equal to 4, 6 and 7.

Based upon these overall results, software company MTM Project srl will proceed to integrate the outcome-oriented process monitoring service provided by ORANGE into CRM.

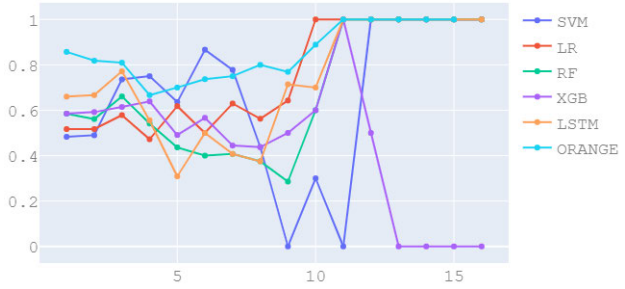


FIGURE 8. AUC on the outcome prediction problem of CVR across different prefix lengths.

V. CONCLUSION

Boosted by the recent application of computer vision approaches in predictive process mining [10], [30], we propose a novel deep learning approach for outcome-oriented predictive process monitoring. This approach founds on the idea of transforming features of traces into pixels of images so that deep learning methods designed for computer vision can be used to process the imagery representation of the traces. In this study, the image encoding of traces is done by modeling the across-feature data continuity so that similar trace features can be mapped onto neighbour pixels. This allows us to construct images of traces, in which possible spatial patterns (e.g. edges, shapes, shading changes) arise. In fact, thanking the embed spatial structure of this imagery representation of traces, we can properly train a 2D CNN that predicts the outcome of an ongoing trace by taking full advantage of convolution and pooling operations.

Experimental results on several real-world event logs confirm the accuracy of the proposed approach compared to various competitors recently illustrated in [5]. They also prove the viability of ORANGE as a means to realize an Intelligent Assistant for CRM – the CVM realized by MTM Project srl, to support sales agents in managing their negations. ORANGE can aid in smartly monitoring the outcome of the ongoing negotiations by highlighting the most promising ones (the negotiations that are candidate to be successfully completed by the sale agent).

Although the experiments have proved the effectiveness of ORANGE in various outcome prediction problems, the proposed method suffers from a few limitations. One limitation of ORANGE is the lack of prescription and explanation with predictions. We plan to explore the use of prescriptive learning theories, such as those investigated in [4], [50], in order to enrich the proposed learning approach with guidelines that describe what to do to achieve specific outcomes and how to expand the model by integrating possible reactions to prediction-based alerts. A further limitation is that ORANGE, similarly to competitors, does not implement any solution to deal with the imbalanced condition that occurs in various outcome prediction problems. To overcome this limitation, we plan to extend the proposed method by implementing trace augmentation techniques, in order to augment the set of training traces with new traces produced to achieve the balance in the learning stage.

As an additional future work, we plan to extend the investigation of the sensitivity of the proposed method to the multiple schemes for the feature vector construction, which are described in [5]. Another interesting research direction is that of extending the proposed approach in a streaming setting with the training performed continuously as new events are logged (in running or new traces). This will require the definition of a streaming framework to update the predictive model as new events are collected and deal with concept drift happening as new events (unobserved before) appear in the process. Finally, transfer learning mechanisms may be also explored to reuse a model developed for business process as the starting point for a model on a new process.

ACKNOWLEDGMENT

The authors thank Luca De Rose for his support in implementing the algorithm to transform feature vector examples into 2D images into Python 3.7.

REFERENCES

[1] F. M. Maggi, C. Di Francescomarino, M. Dumas, and C. Ghidini, "Predictive monitoring of business processes," in *Advanced Information Systems Engineering*, M. Jarke, J. Mylopoulos, C. Quix, C. Rolland, Y. Manolopoulos, H. Mouratidis, and J. Horkoff, Eds. Cham, Switzerland: Springer, 2014, pp. 457–472.

[2] A. Pika, W. van der Aalst, M. Wynn, C. Fidge, and A. ter Hofstede, "Evaluating and predicting overall process risk using event logs," *Inf. Sci.*, vols. 352–353, pp. 98–120, Jul. 2016.

[3] C. D. Francescomarino, M. Dumas, F. M. Maggi, and I. Teinemaa, "Clustering-based predictive process monitoring," *IEEE Trans. Services Comput.*, vol. 12, no. 6, pp. 896–909, Nov./Dec. 2019.

[4] I. Teinemaa, "Predictive and prescriptive monitoring of business process outcomes," in *Proc. Diss. Award, Doctoral Consortium, Demonstration Track Co-Located 17th Int. Conf. Bus. Process Manage. (BPM)* (CEUR Workshop Proceedings), vol. 2420, B. Depaire, Eds. Vienna, Austria: Springer, 2019, pp. 15–19.

[5] I. Teinemaa, M. Dumas, M. L. Rosa, and F. M. Maggi, "Outcome-oriented predictive process monitoring: Review and benchmark," *ACM Trans. Knowl. Discovery Data*, vol. 13, no. 2, pp. 1–57, 2019.

[6] L. Genga, C. D. Francescomarino, C. Ghidini, and N. Zannone, "Predicting critical behaviors in business process executions: When evidence counts," in *Proc. Bus. Process Manage. Forum (BPM Forum)* (Lecture Notes in Business Information Processing), vol. 360, T. Hildebrandt, Eds. Vienna, Austria: Springer, 2019, pp. 72–90.

[7] W. M. P. van der Aalst, *Process Mining–Data Science in Action*, 2nd ed. Berlin, Germany: Springer, 2016.

[8] A. Santoso, "Specification-driven multi-perspective predictive business process monitoring," in *Enterprise, Business-Process and Information Systems Modeling*. Cham, Switzerland: Springer, 2018, pp. 97–113.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[10] V. Pasquadibisceglie, A. Appice, G. Castellano, and D. Malerba, "Predictive process mining meets computer vision," in *Business Process Management Forum (BPM Forum)*. Cham, Switzerland: Springer, 2020, pp. 176–192.

[11] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," *Sci. Rep.*, vol. 9, no. 1, p. 11399, Dec. 2019.

[12] H. Nguyen, M. Dumas, M. La Rosa, F. M. Maggi, and S. Suriadi, "Mining business process deviance: A quest for accuracy," in *Proc. OTM Confederated Int. Conf. Move Meaningful Internet Syst.* Berlin, Germany: Springer, 2014, pp. 436–445.

[13] A. Leontjeva, R. Conforti, C. Di Francescomarino, M. Dumas, and F. M. Maggi, "Complex symbolic sequence encodings for predictive monitoring of business processes," in *Proc. Int. Conf. Bus. Process Manage.* Cham, Switzerland: Springer, 2016, pp. 297–313.

[14] I. Teinemaa, M. Dumas, F. M. Maggi, and C. Di Francescomarino, "Predictive business process monitoring with structured and unstructured data," in *Proc. Int. Conf. Bus. Process Manage.* Cham, Switzerland: Springer, 2016, pp. 401–417.

[15] D. Grigori, F. Casati, U. Dayal, and M.-C. Shan, "Improving business process quality through exception understanding, prediction, and prevention," in *Proc. 27th Very Large Data Base (VLDB) Conf.*, vol. 1, 2001, pp. 159–168.

[16] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.-C. Shan, "Business process intelligence," *Comput. Ind.*, vol. 53, no. 3, pp. 321–343, Apr. 2004.

[17] M. Castellanos, N. Salazar, F. Casati, U. Dayal, and M.-C. Shan, "Predictive business operations management," in *Proc. Int. Workshop Databases Netw. Inf. Syst.* Berlin, Germany: Springer, 2005, pp. 1–14.

[18] B. Kang, D. Kim, and S.-H. Kang, "Periodic performance prediction for real-time business process monitoring," *Ind. Manage. Data Syst.*, vol. 112, no. 1, pp. 4–23, 2012.

[19] A. Senderovich, C. Di Francescomarino, C. Ghidini, K. Jorbina, and F. M. Maggi, "Intra and inter-case features in predictive process monitoring: A tale of two dimensions," in *Proc. Int. Conf. Bus. Process Manage.* Cham, Switzerland: Springer, 2017, pp. 306–323.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] J. Evermann, J.-R. Rehse, and P. Fettke, "A deep learning approach for predicting process behaviour at runtime," in *Proc. Int. Conf. Bus. Process Manage.* Cham, Switzerland: Springer, 2017, pp. 327–338.

[22] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, "Predictive business process monitoring with LSTM neural networks," in *Proc. Int. Conf. Adv. Inf. Syst. Eng.*, E. Dubois and K. Pohl, Eds. Cham, Switzerland: Springer, 2017, pp. 477–492.

[23] M. Camargo, M. Dumas, and O. G. Rojas, "Learning accurate LSTM models of business processes," in *Proc. Int. Conf. Bus. Process Manage.* (Lecture Notes in Computer Science), vol. 11675. T. T. Hildebrandt, B. F. van Dongen, M. Röglinger, and J. Mendling, Eds. Cham, Switzerland: Springer, 2019, pp. 286–302.

[24] F. Taymouri, M. L. Rosa, S. Erfani, Z. D. Bozorgi, and I. Verenich, "Predictive business process monitoring via generative adversarial nets: The case of next event prediction," in *Proc. Int. Conf. Bus. Process Manage.* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2020.

[25] I. Teinemaa, M. Dumas, A. Leontjeva, and F. M. Maggi, "Temporal stability in predictive process monitoring," *Data Mining Knowl. Discovery*, vol. 32, no. 5, pp. 1306–1338, Sep. 2018.

[26] N. Di Mauro, A. Appice, and T. M. A. Basile, "Activity prediction of business process instances with inception cnn models," in *AI*IA–Advances in Artificial Intelligence*, M. Alviano, G. Greco, and F. Scarcello, Eds. Cham, Switzerland: Springer, 2019, pp. 348–361.

[27] J. Evermann, J.-R. Rehse, and P. Fettke, "Predicting process behaviour using deep learning," *Decis. Support Syst.*, vol. 100, pp. 129–140, Aug. 2017.

[28] L. Lin, L. Wen, and J. Wang, "Mm-pred: A deep predictive model for multi-attribute event sequence," in *Proc. SIAM Int. Conf. Data Mining (SDM)*, T. Y. Berger-Wolf and N. V. Chawla, Eds. Philadelphia, PA, USA: SIAM, 2019, pp. 118–126.

[29] N. Mehdiyev, J. Evermann, and P. Fettke, "A multi-stage deep learning approach for business process event prediction," in *Proc. IEEE 19th Conf. Bus. Informat. (CBI)*, vol. 1, Jul. 2017, pp. 119–128.

[30] V. Pasquadibisceglie, A. Appice, G. Castellano, and D. Malerba, "Using convolutional neural networks for predictive process analytics," in *Proc. Int. Conf. Process Mining (ICPM)*, Jun. 2019, pp. 129–136.

[31] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.

[32] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[33] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[34] G. Castellano, C. Castiello, C. Mencar, and G. Vessio, "Crowd detection in aerial images using spatial graphs and fully-convolutional neural networks," *IEEE Access*, vol. 8, pp. 64534–64544, 2020.

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[36] G. E. Hinton, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[37] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 1096–1103.

[38] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
[39] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10. Cambridge, MA, USA: MIT Press, 1995.
[40] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 8609–8613.
[41] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *Proc. Int. Conf. Artif. Neural Netw. (ICANN)*. Springer, 2010, pp. 92–101.
[42] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst.* New York, NY, USA: Curran Associates, 2011, pp. 2546–2554.
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[44] F. Mannhardt. (2016). *Sepsis Cases-Event Log. 4tu.Researchdata. Dataset*. [Online]. Available: https://doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460
[45] B. van Dongen. 2011. *Real-Life Event Logs-Hospital Log. 4tu.Researchdata. Dataset*. [Online]. Available: https://doi.org/10.4121/uuid:d9769f3d-0ab0-4fb8-803b-0d1120ffcf54
[46] (2012). *Bpi challenge 2012. 4Tu.Researchdata. Dataset*. [Online]. Available: https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
[47] D. Levy. (2014). *Production Analysis With Process Mining Technology. 4Tu.Researchdata. Dataset*. [Online]. Available: https://doi.org/10.4121/uuid:68726926-5ac5-4fab-b873-ee76ea412399
[48] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, Jul. 1997.
[49] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
[50] I. Teinemaa, N. Tax, M. de Leoni, M. Dumas, and F. Maria Maggi, "Alarm-based prescriptive process monitoring," 2018, *arXiv:1803.08706*. [Online]. Available: http://arxiv.org/abs/1803.08706

VINCENZO PASQUADIBISCEGLIE graduated in computer science from the University of Bari Aldo Moro, Italy, in December 2017, where he is currently pursuing the Ph.D. degree with the Department of Computer Science. His Laurea thesis was in the area of computational intelligence. He was involved in a regional research project on process mining. His main research interests include process mining, deep learning, big data analytics, and computational intelligence. He is member of the IEEE Task Force on Process Mining. He received the IET's Vision and Imaging Award ICDP in 2019.

ANNALISA APPICE received the Ph.D. degree in computer science from the University of Bari Aldo Moro, Italy. She was a Visiting Researcher with the University of Bristol, U.K., and the Jozef Stefan Institute, Slovenia. She is currently an Associate Professor with the Department of Computer Science, University of Bari Aldo Moro. She has participated with the Organization (Co-Chair) of several workshops, included the First International Workshop on Leveraging Machine Learning in Process Mining (ML4PM), ICPM, in 2020. She has published more than 135 papers in international journals and conferences on these topics. Her current research interests include data mining with spatio-temporal data, data streams, and event logs. She is a member of the IEEE Task Force on Process Mining. She served as the Program Co-Chair for ECML-PKDD in 2015 and ISMIS in 2017. She serves as the Program Chair for DS in 2020 and the Journal Track Chair for ECML-PKDD. She is a member of the editorial board of *Data Mining and Knowledge Discovery (DAMI)* and *Journal of Intelligent Information Systems (JIIS)*.

GIOVANNA CASTELLANO (Member, IEEE) is currently an Associate Professor with the Department of Computer Science, University of Bari Aldo Moro, Italy, where she is also a Coordinator with the Computational Intelligence Laboratory, Computer Science Department. She has published more than 200 papers in international journals and conference proceedings on these topics. Her research interests include computational intelligence and computer vision. She is a member of the IEEE Computational Intelligence Society, the EUSFLAT Society, and the INDAM-GNCS Society. She was a Co-Organizer with the fourth EUSFLAT European Summer School on Fuzzy Logic and Applications (SFLA) in 2018. She serves as the General Chair for the 2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (IEEE-EAIS) in 2020. She also serves as an Associate Editor for *Information Sciences*, *Evolving Systems*, the *International Journal of Intelligent Systems*, the *International Journal of Systems, Control and Communications*, and the *International Journal of Knowledge-Based and Intelligent Engineering Systems*.

DONATO MALERBA (Member, IEEE) was the Board of Directors with the Big Data Value Association and the Partnership Board, PPP Big Data Value. He is currently a Full Professor and the Director with the Department of Computer Science, University of Bari Aldo Moro, Italy. He is also with the CINI Laboratory on Big Data. He was responsible for the local research unit of several European and national projects. He has published more than 300 papers in international journals and conference proceedings. His research interests include machine learning, data mining, and big data analytics and their applications. He is a member of the IEEE Task Force on Process Mining. He received the IBM Faculty Award in 2004. He served as the Program (Co-) Chair for IEA-AIE in 2005, ISMIS in 2006, SEBD in 2007, and ECMLPKDD in 2011. He also served as the General Chair for ALT/DS in 2016. He serves on the editorial board of several international journals.

GIUSEPPE MODUGNO graduated in mechanical engineering from the University of Bari Aldo Moro, Italy, in June 2006. He received the M.B.A. degree from the University of Bologna, in 2008. His thesis was on application of project management techniques to the construction of a diagnostic train for the Madrid Metro. He is currently a Sole Administrator with MTM Project srl, Innovative Italian SME. His scientific publications are Training in Virtual Reality: The Magna Getrag Case, in May 2019, and a process mining approach to predict outcome of worker training done through a virtual learning environment published by the IEEE, in May 2020. His first involvement in research activities dates back to 2002 in railway. He is also involved in a regional research project in virtual reality and artificial intelligence.