sciendo

JAISCR, 2022, Vol. 12, No. 4, pp. 255 – 266
10.2478/jaiscr-2022-0017

DETECTING ANOMALIES IN ADVERTISING WEB TRAFFIC WITH THE USE OF THE VARIATIONAL AUTOENCODER

Marcin Gabryel[1,*], Dawid Lada[2], Zbigniew Filutowicz[3], Zofia Patora - Wysocka[4], Marek Kisiel - Dorohinicki[5], Guang Yi Chen[6]

[1]*Department of Intelligent Computer Systems, Częstochowa University of Technology, al. Armii Krajowej 36, 42-200 Częstochowa, Poland*

[2]*Spark Digitup, 31-060 Krakow, Poland*

[3]*Institute of Information Technologies, University of Social Sciences, ul. Sienkiewicza 9, 90-113 Lodz*

[4]*Management Department, University of Social Science, ul. Sienkiewicza 9, 90–113 Lodz, Poland*

[5]*Institute of Computer Science, AGH University of Science and Technology, 30-059 Krakow, Poland*

[6]*Department of Computer Science and Software Engineering Concordia University, Montreal, Quebec, Canada H3G 1M8*

[*]*E-mail: marcin.gabryel@pcz.pl*

Submitted: 2nd April 2022; Accepted: 12th October 2022

Abstract

This paper presents a neural network model for identifying non-human traffic to a website, which is significantly different from visits made by regular users. Such visits are undesirable from the point of view of the website owner as they are not human activity, and therefore do not bring any value, and, what is more, most often involve costs incurred in connection with the handling of advertising. They are made most often by dishonest publishers using special software (bots) to generate profits. Bots are also used in scraping, which is automatic scanning and downloading of website content, which actually is not in the interest of website authors. The model proposed in this work is learnt by data extracted directly from the web browser during website visits. This data is acquired by using a specially prepared JavaScript that monitors the behavior of the user or bot. The appearance of a bot on a website generates parameter values that are significantly different from those collected during typical visits made by human website users. It is not possible to learn more about the software controlling the bots and to know all the data generated by them. Therefore, this paper proposes a variational autoencoder (VAE) neural network model with modifications to detect the occurrence of abnormal parameter values that deviate from data obtained from human users' Internet traffic. The algorithm works on the basis of a popular autoencoder method for detecting anomalies, however, a number of original improvements have been implemented. In the study we used authentic data extracted from several large online stores.

Keywords: anomaly detection, web traffic, ad fraud, variational autoencoder

1 Introduction

One of the main types of online crime and fraud has become so-called traffic fraud. These procedures involve fraudulently increasing online advertising revenue by automatically generating page views, clicks or filling out online forms, which bring fraudsters real financial revenue and profit them by generating losses in competing businesses. Globally, ad fraud is causing multi-billion dollar losses to the advertising industry [6, 7], as many website owners do not have sufficient safeguards against this kind of fraudulent activity.

The advertising system used for promoting websites relies on the cooperation of three main parties, i.e. publishers, advertisers and affiliate networks. Publishers provide network services to users and in the process provide resources for advertising traffic. Traffic is generated when a user visits their sites. A user's visit to a publisher's site creates an opportunity for one or more ads to be displayed. The advertiser buys traffic so as to deliver its ads to the website visitor. Affiliate networks are the intermediary between publishers and advertisers. The publisher sells ad space to interested advertisers, and does it through an intermediary – the affiliate network. Intermediaries also offer support for marketing activities for advertisers, such as affiliate partners, website owners and product comparison sites.

All kinds of fraud and ad fraud activities cause losses primarily to the advertiser who allocates certain financial resources to an advertising campaign hoping to attract more users. Publishers and intermediaries are usually billed on a pay-per-performance model. Evaluating the effectiveness of online advertising can be done in several forms including CPC – cost-per-click, CPM – cost-per-mile, CPL – cost-per-lead or CPS – cost-per-sale. The most common frauds detected in online advertising involve performance ads, i.e. ads billed for the number of ad clicks generated. In addition to "ad clicks," the effect of fraudulent publishers' actions can also include serving fake leads. Such leads are generated by filling out forms with abnormal data or persons' data obtained from another source. People whose data is entered into such online form are often unaware of this and have not given their consent to the processing of their data. In extreme cases, there are also fictitious sales of products from advertisers' online stores resulting in payment of unjustified commissions. Most of such procedures are carried out by specially developed software – bots.

Fraud also occurs in the case of carrying out automatic scanning of websites (scraping). This is done most often in order to gain information about the prices offered by competitors. Knowing the pricing offered by other sellers allows you to adequately adjust the pricing of your own products to set a competitive price and attract more customers. Websites also contain valuable copyrighted content, and fraudulent scraping allows their content to be downloaded without the permission of website authors.

In addition to fraudulent publishers and intermediaries, many types of abuse could occur through attempts to use unfair competition. Automated software or even manual clicking of your competitor's ads allows you to ensure better positioning of your ads and lower the cost per click of your own ad. These actions consume the advertising budget of competitors and do not bring them expected performance conversions (increased sales results), consequently reducing the actual scale of the advertising campaign.

Most advertisers do not have mechanisms to automatically monitor in real time the behavior of users on their website. In fact, it is possible to retrieve certain data concerning the user behavior during their visit and then interpret it accordingly. This could make it possible to identify and notify system administrators of the occurrence of unusual data that have the character of anomalies. Such anomalies can be observed during visits to the site made by automated software, i.e. bots. This software, in order to emulate human behavior, must interact with the website. As a result, it can automatically fill out and submit an online form, perform ad clicks or download page content. However, anomaly detection requires implementation of additional software to track user behavior on a website in detail. This can be enabled by an appropriate JavaScript running directly in the browser.

Analyzing and searching for unusual data without the support of appropriate detection algorithms is highly labor-intensive for humans. It often requires reviewing hundreds of thousands of records. Their number increases as the popularity of the

website increases and consequently, the number of visitors to the site increases. Searching the data manually is by no means cost-effective and a lot of information can be overlooked. Usually, the detection of unusual data may occur long after the fact, or such data may actually go unnoticed. There are a number of off-the-shelf security solutions available to help distinguish the presence of a bot from a human with a certain degree of accuracy (e.g., [1] or [10]). However, these services are offered by third-party companies, their algorithms are not freely available and, what is more, they require a direct response from the user or a long time to conduct an analysis. Unlike the above-mentioned solutions, the approach proposed in this paper makes it possible to quickly identify a visit made to a website by a bot. An additional benefit of using this method is that anomalies may result from human error, for example, a misplaced advertisement on a website. The algorithm also provides an opportunity to detect errors related to rapidly emerging successive versions of browsers and their backward compatibility. The information received from the model will allow the online store administrator to react quickly to incidents that occur on the website. Identifying a visit as unusual, not originating from a human, can also bring real financial benefits. Identified ad frauds can be the basis for questioning the billing with the affiliate network or publisher indicating ineffectiveness of the advertisement in detected cases.

This paper will present a model of an artificial neural network that allows automatic detection of anomalies in Internet traffic in real time. The paper consists of several sections. Section 2 introduces similar solutions for detecting anomalies using autoencoders and VAEs. In the next two sections, i.e. Sections 3 and 4, a neural network model with autoencoder and VAE structure is described and the proposed solution is explained. Section 5 presents the results of experimental work showing the effectiveness of the new method, and finally, Section 6, summarizes the work on the model to date and our plans for its further development.

2 Similar solutions

There are many studies in the literature that use deep autoencoder neural networks [2, 17] to detect anomalies among data, data streams, images and others. One of the first papers that presented using VAE for anomaly detection was [16]. There, a method was proposed to detect anomalies using reconstruction probabilities obtained from a variational autoencoder. The authors of the publication [4] proposed a deep learning system for detecting cyber intrusions by solving the problem of identifying network attacks associated with a large number of security vulnerabilities in computer systems. In the model used, special attention was paid to ensure that the deep autoencoder avoids over-fitting as well as the local optimum. Possible cyber intrusion detection capabilities for computer systems were also studied by the authors of [12]. They proposed the use of stacked ensembles consisting of neural network (SNN) and autoencoder (AE) models improved by a new approach to optimizing hyperparameters. Then, in [5] the problem of network anomaly detection was solved using a variational autoencoder. Anomaly detection can also be useful in fields such as geochemistry. The authors of [8] train an autoencoder network to encode and reconstruct populations of geochemical samples with unknown complex multivariate probability distributions. During the training, small probability samples contribute little to the autoencoder network. These samples can be recognized by the trained model as anomalous samples due to their comparatively higher reconstructed errors. In the paper [3], the autoencoder was used for a slightly different purpose. It was able to effectively learn already noisy data and to remove outliers. This makes it possible to take advantage of a remarkable generalization ability of this neural network structure. The authors of [11], on the other hand, studied the method of detecting patterns of anomalies in the data stream based on the autoencoder, among others, and compared it with other methods (the Isolation Forest and Local Outlier Factor). In the paper [13], a hybrid CNN-VAE architecture was used for trajectory classification and anomaly detection. Its authors managed to identify some of the important traffic anomalies, such as vehicles not following the lane along which they are driving, sudden changes in speed, sudden interruption of vehicle movement, and vehicles moving in the wrong directions.

A new approach, called S2-VAE, for detecting abnormal events from video sequences was proposed in [14]. The S2-VAE model used consists

of two proposed neural networks: Stacked Fully Connected Variational AutoEncoder (SF-VAE) and Skip Convolutional VAE (SC-VAE).

The article [15] presents a deep support vector data description based on a variational autoencoder (Deep SVDD-VAE). In the proposed model, the VAE is used to reconstruct the input instances, while a spherical discriminative boundary is learned with the latent representations simultaneously based on the SVDD.

A large summary of anomaly detection methods can be found in the paper [17]. The authors defined six challenges associated with the problem and assigned to them well-known methods, including autoencoders, to solve them. On the other hand, a framework was proposed in [21], where one model is able to detect three different types of anomalies: out-of-distribution samples, adversarial samples and noise samples.

3 Algorithm used

3.1 Autoencoder

Autoencoders are artificial neural network structures capable of learning efficient representations of input data without a teacher. It is characteristic of a correctly learned autoencoder that it can reproduce at the output without error what is given at its input. An important feature is that one of the middle layers contains a very small number of neurons and its task is to reduce the dimension of the data fed at the input. It is located at the end of the encoder, the part of the network that includes the first layers connecting it to the said smallest layer. The compressed data is then passed to the second part of the autoencoder (decoder), whose task is to decode the data from the smallest layer and reconstruct it at the output. Such a neural network has generalization capabilities because the data is not passed directly from the input to the output. A perfectly learned autoencoder recognizes the input data by returning identical output values. There is then a small reconstruction error, i.e. the difference between the expected (input) values and the values obtained during data reconstruction. The data, with which the autoencoder has not been learned, fed at the input will result in the generation of a large value of the reconstruction error. Verification of the value of the obtained error gives the possibility to make classifications of data into data known or unknown to the autoencoder (anomalies).

For a given training set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \in R^m$ where x_i is an m-dimensional feature vector, N — the number of samples, the autoencoder structure can be described as follows. The encoder maps the input vector $\mathbf{x}_i$ on the latent representation $\mathbf{z} \in R^n$ by using f_Φ according to the formula:

$$\mathbf{z} = f_\Phi(\mathbf{x}_i) = s(\mathbf{W}\mathbf{x}_i + \mathbf{b}), \tag{1}$$

where $\mathbf{W} \in R^{(m \times n)}$ is a set of weights, n is the number of units in the latency layer $\mathbf{z}$, $\mathbf{b} \in R^n$ is a bias vector, Φ is a set $\{\mathbf{W}, \mathbf{b}\}$, and $s(\cdot)$ is the adopted activation function (sigmoid function) determined by the formula:

$$s(t) = \frac{1}{1 + exp^{-t}}. \tag{2}$$

The decoder maps back values z_i obtained in the latency layer on the output vector $\mathbf{y}_i \in R^n$ as in the following form:

$$\mathbf{y}_i = g_\Theta(\mathbf{x}_i) = s(\acute{\mathbf{W}}\mathbf{z} + \acute{\mathbf{b}}), \tag{3}$$

where $\acute{\mathbf{W}} \in R^{(n \times m)}$ are the weights, $\acute{\mathbf{b}} \in R^m$ is a bias vector and $\Theta = \{\acute{\mathbf{W}}, \acute{\mathbf{b}}\}$.

The training of the autoencoder consists in minimizing the difference between the input $\mathbf{x}_i$ and output $\mathbf{y}_i$. To this end a loss function is calculated, which is expressed in the following formula:

$$L(\mathbf{x}_i, \mathbf{y}_i) = \|\mathbf{x}_i - \mathbf{y}_i\|^2 = \|\mathbf{x}_i - s(\acute{\mathbf{W}} s(\mathbf{W}\mathbf{x}_i + \mathbf{b}) + \acute{\mathbf{b}})\|^2. \tag{4}$$

The aim of the learning is to find the optimum values of parameters Φ and Θ so as to minimize the reconstruction error between the input and the output for the whole training set:

$$\Phi, \Theta = \underset{\Phi,\Theta}{\operatorname{argmax}} L(\mathbf{x}, \mathbf{y}). \tag{5}$$

The autoencoder presented above has been considered for continuous data $\mathbf{x}$.

3.2 Variational autoencoder

Variational Autoencoder (VAE) [18] is a probabilistic autoencoder, which means that it generates partially random results, even after the model has been learned. It can generate new samples that resemble the data included in the learning set.

The VAE architecture resembles an autoencoder – it has two clearly separated parts: an encoder and a decoder. However, a significant modification is found here. The encoder generates mean coding μ and a standard deviation σ. If by using $\mathbf{x}$ we designate the input and by using z we designate the latent representation, then the VAE will consist of two parts: 1) an encoder $q_\Phi(\mathbf{z}|\mathbf{x})$ operating as a recognition model representing the approximation to the intractable true posteriori $p_\Theta(\mathbf{z}|\mathbf{x})$, and 2) a decoder $p_\Theta(\mathbf{x}|\mathbf{z})$ as generative model to generate new data with latent representation $\mathbf{z}$. With the data set $\mathbf{X}$, the purpose of the learning of the VAE is to find the maximum likelihood $\sum_{i=1}^{n} \log p_\Theta(\mathbf{x}_i)$ in relation to the parameters of the encoder and the parameters of the decoder Θ, where $\log p_\Theta(\mathbf{x}_i)$ can be expressed with the formula:

$$\log p_\Theta(\mathbf{x}_i) = D_{KL}\left(q_\Phi(\mathbf{z}|\mathbf{x}_i)\|p_\Theta(\mathbf{z})\right) + L(\Theta;\Phi;\mathbf{x}_i), \tag{6}$$

where $p_\Theta(\mathbf{z})$ is the prior over latent variables and $D_{KL}(\cdot)$ is the Kullback-Leiber (KL) divergence [19]. Expression $L(\Theta;\Phi;\mathbf{x}_i)$ is called the evidence variational lower bound (ELBO) on the marginal likelihood of sample $\mathbf{x}_i$. Since expression $D_{KL}(\cdot)$ is nonnegative, formula (6) can be expressed as follows:

$$\log p_\Theta(\mathbf{x}_i) \geq L(\Theta;\Phi;\mathbf{x}_i). \tag{7}$$

Since $\log p_\Theta(\mathbf{x}_i)$ is intractable, then the ELBO is maximized instead so as to evaluate the maximum likelihood $\log p_\Theta(\mathbf{x}_i)$. The ELBO can be represented as follows:

$$\begin{aligned} L(\Theta;\Phi;\mathbf{x}_i) = &-D_{KL}\left(q_\Phi(\mathbf{z}|\mathbf{x})\|p_\Theta(\mathbf{z})\right) \\ &+\mathbb{E}_{q_\Phi(\mathbf{z}|\mathbf{x}_i)}\left[\log p_\Theta(\mathbf{x}_i|\mathbf{z})\right]. \end{aligned} \tag{8}$$

Part of formula $\mathbb{E}_{q_\Phi(\mathbf{z}|\mathbf{x}_i)}$ is treated as an expected reconstruction error between the input data which require a random latent variable $\mathbf{z}$ sampling from the approximate posterior $\log p_\Theta(\mathbf{x}_i)$. The backpropagation of errors cannot work with a random variable $\mathbf{z}$. When $q_\Phi(\mathbf{z}|\mathbf{x}_i) \sim \mathcal{N}(\mathbf{z};\mu;\sigma^2)$, every random variable $\mathbf{z}_i$ can be represented as $\mathbf{z}_i^l = \mu_i + \sigma_i \cdot \mathcal{N}(0,\mathbf{I})$, and formula (8) takes the form:

$$\begin{aligned} L(\Theta;\Phi;\mathbf{x}_i) = &-D_{KL}\left(q_\Phi(\mathbf{z}|\mathbf{x}_i)\|p_\Theta(\mathbf{z})\right) \\ &+\frac{1}{L}\log p_\Theta(\mathbf{x}_i|\mathbf{z}_i^l). \end{aligned} \tag{9}$$

To optimize the KL-divergence between $q_\Phi(\mathbf{z}|\mathbf{x})$ and $p_\Theta(\mathbf{z})$ under the assumption that $p_\Theta(\mathbf{z})$ follows the Gaussian distribution, the encoder estimates the parameter vectors of the Gaussian distribution $q_\Phi(\mathbf{z}|\mathbf{x})$: mean and standard deviation , i.e. $q_\Phi(\mathbf{z}|\mathbf{x} \sim \mathcal{N}(\mathbf{z};\mu;\sigma^2)$. The explicit expression for KL-divergence between $q_\Phi(\mathbf{z}|\mathbf{x})$ $p_\Theta(\mathbf{z})$ can be simply written as:

$$\begin{aligned} &-D_{KL}\left(q_\Phi(\mathbf{z}|\mathbf{x}_i)\|p_\Theta(\mathbf{z})\right) \\ &= -D_{KL}\left(\mathcal{N}(\mu_i;\sigma_i^2)\right)\|\mathcal{N}(0,\mathbf{I}) \\ &= \frac{1}{2}\sum\sum_{j=1}^{z_d}\left(1+\log\left(\sigma_i^{j^2}\right)-\mu_i^{j^2}-\sigma_i^{j^2}\right), \end{aligned} \tag{10}$$

where z_d is dimension $\mathbf{z}$. Then, the VAE objective function for $\mathbf{x}_i$ is:

$$\begin{aligned} &L(\Theta;\Phi;\mathbf{x}_i) \\ &= \frac{1}{2}\sum\sum_{j=1}^{z_d}\left(1+\log\left(\sigma_i^{j^2}\right)-\mu_i^{j^2}-\sigma_i^{j^2}\right) \\ &\quad +\frac{1}{L}\log p_\Theta(\mathbf{x}_i|\mathbf{z}_i^l). \end{aligned} \tag{11}$$

This function is optimized by backpropagation of errors, where the parameters Θ and Φ are learned simultaneously.

4 Proposed solution

Each visit to a website can be identified as a visit by a human or a bot by analyzing data acquired by monitoring interactions with elements of the website and collecting information about the browser, operating system and computer hardware used by the user. This monitoring is carried out by a JavaScript that runs when all the content of a website has been loaded. Software for generating a browser's unique fingerprint operates based on a similar principle [20]. The browser, when normally used by a human, during a single page view of a website generates and allows downloading of a complete set of parameter values. An overwhelming number of data have similar values and no empty values. However, there are atypical entries to the website made by various types of software that create so-called non-human traffic. Such data can come, for example, from software that tries to ineptly emlate the behavior of a human user, but it can also come from a new version of a browser that no longer supports some functionality. During such accesses to the website, most often the monitored

parameter values differ significantly from those collected so far, and there are often numerous gaps in the values of the collected parameters or errors are generated during their acquisition. The main purpose of the solution proposed in this paper is to detect such data-anomalies, which is normally done manually by analysts during tedious review of the data flowing from the monitored site.

4.1 The algorithm

The starting point for the proposed model is the classical VAE structure. The process of learning this neural network is identical to the learning of an autoencoder. The prepared model is learnt correct training data without anomalies. The learned model should return a small loss error at the output when given a correct sample at the input and a large loss error when given an abnormal sample at the input. The diagram of the VAE in question is shown in Figure 1. Since the data used is of categorical type, the input data is converted to zero-one form by a one-hot-encoder. Similarly, softmax layers are applied on the output for each feature separately. This allows for a simpler representation of one-hot input data.

The data is of class $\Omega = \omega_1, \ldots, \omega_C$, where C is the number of classes. The input data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, where $\mathbf{x}_i$ is an m-dimensional feature vector, and N is the number of samples.

The preparation of the model for classification proceeds in several steps:

1. Preparing the VAE model. Determining the dimension z_d of the latency layer $\mathbf{z}$.
2. Dividing data into correct data marked $\mathbf{x}_N$ and abnormal data (anomalies) - $\mathbf{x}_A$. By analogy, sets of two classes will be created: $\Omega_N \in \{\omega_{N_1}, \omega_{N_2}, \ldots, \omega_{N_C}\}$ for correct samples and $\Omega_A \in \{\omega_{A_1}, \omega_{A_2}, \ldots, \omega_{A_C}\}$ for abnormal ones.
3. The VAE is trained using the correct training data $\mathbf{x}_N$ verifying the correctness of the learning process using validation data.
4. Cutting off the decoder. The model consists only of the encoder with the latency layer $\mathbf{z}$.
5. All the training data is fed on the input. For each sample $\mathbf{x}_i$ pairs are calculated (μ_{ij}, σ_{ij}), $j = 1, \ldots, z_d$ as well as the value $\mathbf{z}_i$.
6. For each set of obtained results of the correct samples $(\mathbf{z}_i, \omega_{N_k})$ and the abnormal ones $(\mathbf{z}_i, \omega_{A_k})$ centroids τ_{N_k} and τ_{A_k} are accordingly calculated, where $k = 1, \ldots, C$.
7. Determining the values of thresholds υ_{N_k} and υ_{N_k} for each of the centroids τ_{N_k} and τ_{A_k}.

Centroids τ_{N_k} and τ_{A_k} along with the threshold values υ_{N_k} and υ_{N_k} make it possible to define certain areas within which the data returned from the latency layer will be considered as correct samples or abnormal ones (anomalies), which will be used in the classification process.

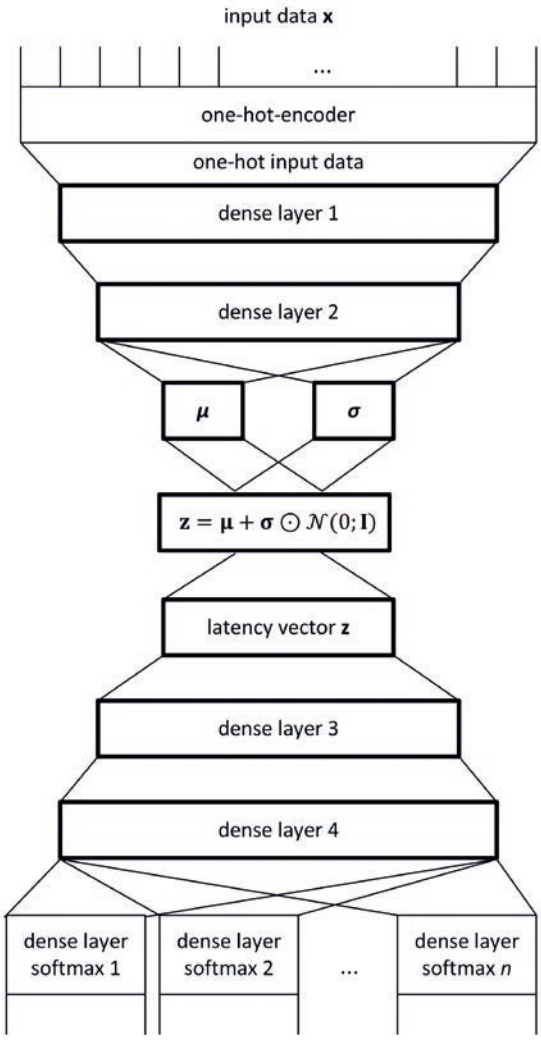


Figure 1. Diagram of the VAE structure used in the proposed method.

When classifying an input sample **x**, you need to perform the following steps:

1. Give the tested sample **x** at the VAE model's input.
2. Calculate value **z**, obtained as the output of the VAE encoder from the latency layer.
3. Calculate the distances to the centroids τ_{N_k} and τ_{A_k} using the defined distance measurement $d(\cdot)$. Return the class $\omega_w \in \{\Omega_N, \Omega_A\}$ belonging to the corresponding centroid τ_{N_k} or τ_{A_k}, to which the distance $d(\cdot)$ is the smallest and is calculated according to the formula:

$$\omega_w = \underset{k=1,\ldots,C}{\operatorname{argmin}} \left(d(\mathbf{z}', \tau_{N_k}); d(\mathbf{z}', \tau_{A_k})\right) \Leftrightarrow \\ d(\mathbf{z}', \tau_{N_k}) < \upsilon_{N_k}; d(\mathbf{z}', \tau_{A_k}) < \upsilon_{A_k} \tag{12}$$

A schematic representation of the model prepared for classification is shown in Figure 2.

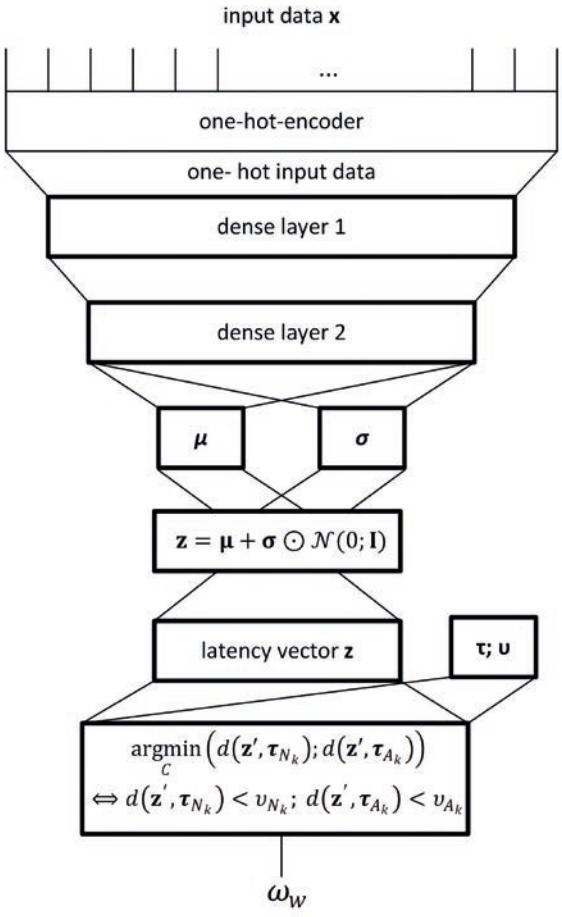


Figure 2. Diagram of the anomaly classification process.

5 Experimental work

The data used for the experiments described in this article were obtained from 9 large online stores and come from the process of monitoring web traffic related to the service of online advertising. In the course of monitoring the ways in which the websites were accessed, two groups of visits to the websites were identified, i.e. those generated at the moment of entering the website after clicking on the advertisement (for the models of CPC and CPM type ad billing) and those generated at the moment of filling in the data in the contact form (for the models billed on CPL and CPS bases). Due to the significant differences in the number of monitored parameters in each of these cases, each of these groups is studied separately and a separate model for anomaly detection should be prepared for each of them. This paper presents a model dedicated to detecting anomalies in web traffic generated when a form is being filled out.

During a single visit, 49 different parameters whose values are retrieved from the web browser by a specially prepared JavaScript are monitored. The parameters include: type of client (indicated as "client_id"), type of device ("is_mobile", "touch_enabled", "virtual_machine"), source of entry to the page ("source_id"), time spent visiting the page ("first_req_st2_close_seconds"), information whether the page was opened in a frame ("in_frame"), information about the tampered data ("os_t_platformOK", "os_t_touchOK", "screen_tampering", "language_tampering", "br_t_evalOk", "br_t_subOk"), information about the screen ("scr_num"), ISP and connection data ("proxy_flag", "ip_type", "isp_is_suspicious", "is_known_crawler_bot"), number of pages visited per session ("history_len"), sales information ("had_sale"), browser information ("browser_type_id", "notif_permission", "chrome_prop_not_set", "browser_type_name"). Additionally monitored parameters include user behavior, i.e. the number of fields to be filled in without pressing a key, the number of text changes made, the number of the "Tab" keys is pressed, the number of mouse pointer clicks on controls, the number of data copied from the clipboard, the number of times individual mouse keys are pressed, the number of keys pressed on the keyboard, and information about mouse

pointer movements (”counter”, ”has_ws_mouse”, ”has_ws_scroll”, ”mm_unique_points”). The parameters are described in more detail in papers [9] and [20].

The data were assigned to one of three classes whose labels were given the names: ”ok”, ”fake” and ”copy”. These classes were defined at the discretion of expert and they define the quality of access to the website. The class ”ok” means that traffic to the site is normal, coming from a human, ”fake” - is a short visit to the website, and ”copy” - the access to the site was normal, but the data on the form was entered by copying it from the clipboard. The vast majority of data obtained during the monitoring of the website is easily classified and recognized as correct. However, a sizable portion of the data is incomplete, lacking all parameter values or having abnormal, non-standard values. These are treated as abnormal data, i.e. anomalies.

A total of 196701 records were selected for the study. The number of correct samples, i.e., those with no anomalous features, is 2742. The remaining have a varying number of abnormal feature values. During the experiments that followed, the focus was on finding those that had a certain number of abnormal feature values. Series of attempts were made to classify anomalies starting with an attempt of a minimum of one and ending with those with a minimum of twenty abnormal values.

During the course of the experiments, many different VAE and autoencoder structures were examined. The most optimal turned out to be the models with 5 layers in the configuration of 256-64-latency-64-256. In addition to these 5 layers, to ensure their correct operation, the neural networks had several additional modules which were used for processing the input data and for obtaining the corresponding values at the output. Since in most cases the input data is categorical, the input data is preprocessed by a one-hot-encoder. This module is designed to convert the integer values of 49 features into form of digital 0s and 1s. For the selected experimental data, a vector of 1676 numbers containing ”0” and ”1” is given onto the input. The learning objective of the prepared model is to reproduce an identical vector on the output, which is why the last layer consists of 49 parallel layers of neurons with the softmax activation function. This structure makes it possible to generate a single ”1” for each feature of the input vector.

In the experiments conducted, four models were tested:

1. Autoencoder (AE) - a detailed description of the application of autoencoder to detect anomalies in Internet traffic can be found in [20]. The model recognizes samples as anomalies when the loss error exceeds a certain threshold value.
2. Variational Autoencoder (VAE) – in this case, the classification algorithm is identical to that of AE except for the fact that a variational autoencoder was used.
3. The presented method using the Manhattan distance in formula (12).
4. The presented method using the Euclidean distance in formula (12).

In each case, additionally, the effectiveness of the models was examined for different latency dimensions z_d. The results of the experiments can be found in Tables 1-4. The successive rows show the results of anomaly detection as measured by F1 score. Anomalies with a different number of abnormal values occurring in a single record were investigated – from a minimum of one value to a minimum of twenty. The successive columns provide information on: the minimum number of abnormal parameters, the number of classified samples and the F1 score values obtained by the four models tested. As can be seen, the best performing models were those with a low latency dimension ($z_d = 2$ lub $z_d = 3$). Our proposed model with the Euclidean distance has proven to be the best for each latency dimension, and in particular for $z_d = 2$. Table 5 shows the F1 scores for the process of learning the models during the conducted experiments. They include the results obtained by the entire autoencoder structure and VAE separately for the training and test data. Figure 3 shows the visualization of latency space for $z_d = 3$ obtained by the best of the models. On the left there are graphs showing the correct samples divided into 3 classes, and on the right additionally there are anomalies marked with black dots. Clear groups of each class can be noticed. The records that have anomalies noticeably stand out from the correct samples.

Table 1. F1 score results for the models tested with latency dimension $z_d = 2$.

Minimum number of abnormal parameters	Number of samples	F1 score			
		AE	VAE	Proposed model - Manhattan distance	Proposed model - Euclidean Distance
1	193959	0.7820	0.8092	0.9980	0.9999
2	165177	0.8260	0.8069	0.9984	0.9999
3	125520	0.8630	0.7689	0.9982	0.9999
4	114973	0.8763	0.7615	0.9980	0.9999
5	89528	0.9087	0.8300	0.9987	1.0000
6	65567	0.9767	0.9320	0.9986	1.0000
7	61886	0.9875	0.9378	0.9999	1.0000
8	42158	0.9919	0.9881	1.0000	1.0000
9	6843	0.9604	0.9729	1.0000	1.0000
10	6376	0.9717	0.9893	0.9998	1.0000
11	6101	0.9702	0.9972	0.9998	1.0000
12	5777	0.9687	0.9977	0.9996	1.0000
13	5681	0.9720	0.9979	0.9996	0.9997
14	3812	0.9634	0.9988	1.0000	1.0000
15	2151	0.9464	1.0000	1.0000	1.0000
16	1367	0.9930	0.9993	1.0000	1.0000
17	991	0.9919	1.0000	1.0000	1.0000
18	46	1.0000	1.0000	1.0000	1.0000
19	45	1.0000	1.0000	1.0000	1.0000
20	44	1.0000	1.0000	1.0000	1.0000

Table 2. F1 score results for the models tested with latency dimension $z_d = 3$.

Minimum number of abnormal parameters	Number of samples	F1 score			
		AE	VAE	Proposed model - Manhattan distance	Proposed model - Euclidean Distance
1	193959	0.6449	0.9145	0.9669	0.9989
2	165177	0.6911	0.9369	0.9696	0.9991
3	125520	0.7721	0.9434	0.9901	0.9999
4	114973	0.7859	0.9408	0.9960	0.9999
5	89528	0.8422	0.9395	0.9962	1.0000
6	65567	0.8799	0.9355	0.9956	1.0000
7	61886	0.8871	0.9346	0.9977	1.0000
8	42158	0.8546	0.9206	0.9989	1.0000
9	6843	0.9596	0.8192	0.9947	1.0000
10	6376	0.9643	0.8187	0.9942	1.0000
11	6101	0.9665	0.8242	0.9948	1.0000
12	5777	0.9620	0.8229	0.9944	1.0000
13	5681	0.9628	0.8187	0.9946	1.0000
14	3812	0.9509	0.8580	0.9924	1.0000
15	2151	0.9317	0.8725	1.0000	1.0000
16	1367	0.8998	0.8375	1.0000	1.0000
17	991	0.8751	0.9045	1.0000	1.0000
18	46	0.9890	0.9176	1.0000	1.0000
19	45	1.0000	0.8608	1.0000	1.0000
20	44	1.0000	0.9268	1.0000	1.0000

Table 3. F1 score results for the models tested with latency dimension $z_d = 4$.

Minimum number of abnormal parameters	Number of samples	F1 score			
		AE	VAE	Proposed model - Manhattan distance	Proposed model - Euclidean Distance
1	193959	0.7773	0.9275	0.9797	0.9985
2	165177	0.8127	0.9397	0.9808	0.9994
3	125520	0.8401	0.9346	0.9833	0.9996
4	114973	0.8429	0.9331	0.9423	0.9974
5	89528	0.8844	0.9400	0.9836	0.9998
6	65567	0.9561	0.9394	0.9383	0.9978
7	61886	0.9652	0.9374	0.9809	0.9998
8	42158	0.9826	0.9779	0.9994	0.9999
9	6843	0.9664	0.9196	0.9264	0.9986
10	6376	0.9687	0.9341	0.9994	1.0000
11	6101	0.9719	0.9448	0.9991	0.9999
12	5777	0.9716	0.9474	0.9959	0.9996
13	5681	0.9713	0.9505	0.9949	0.9988
14	3812	0.9586	0.9524	0.9946	1.0000
15	2151	0.9396	0.9593	0.9190	0.9998
16	1367	0.9064	0.9578	0.9947	0.9992
17	991	0.8783	0.9920	0.9952	0.9992
18	46	1.0000	1.0000	1.0000	1.0000
19	45	1.0000	1.0000	1.0000	1.0000
20	44	1.0000	1.0000	1.0000	1.0000

Table 4. F1 score results for the models tested with latency dimension $z_d = 5$.

Minimum number of abnormal parameters	Number of samples	F1 score			
		AE	VAE	Proposed model - Manhattan distance	Proposed model - Euclidean Distance
1	193959	0.6919	0.9320	0.9463	0.9962
2	165177	0.7255	0.9362	0.9451	0.9960
3	125520	0.7539	0.9270	0.9365	0.9968
4	114973	0.7578	0.9233	0.9334	0.9968
5	89528	0.8166	0.9258	0.9386	0.9957
6	65567	0.9025	0.9261	0.9452	1.0000
7	61886	0.9266	0.9261	0.9457	1.0000
8	42158	0.9394	0.9839	0.9245	1.0000
9	6843	0.9329	0.9742	0.9749	1.0000
10	6376	0.9356	0.9852	0.9773	1.0000
11	6101	0.9389	0.9874	0.9811	1.0000
12	5777	0.9403	0.9895	0.9799	1.0000
13	5681	0.9373	0.9882	0.9759	0.9997
14	3812	0.9210	0.9824	0.9778	1.0000
15	2151	0.9461	0.9892	0.9954	1.0000
16	1367	0.9687	0.9875	0.9971	1.0000
17	991	0.9530	0.9924	0.9949	1.0000
18	46	1.0000	1.0000	1.0000	1.0000
19	45	1.0000	1.0000	1.0000	1.0000
20	44	1.0000	1.0000	0.9660	1.0000

Table 5. F1 score obtained for the teaching and testing sequence in the successive experiments.

z_d	Data	F1 score			
		AE	VAE	Proposed model - Manhattan distance	Proposed model - Euclidean Distance
2	Train	0.9929	0.9929	0.9744	0.9929
2	Test	0.4060	0.9948	0.9639	0.9948
3	Train	0.9965	0.9987	0.9674	0.9987
3	Test	0.3412	1.0000	0.9708	1.0000
4	Train	0.9806	0.9920	0.8772	0.9920
4	Test	0.3255	0.9945	0.8302	0.9945
5	Train	0.9965	0.9941	0.8666	0.9941
5	Test	0.3307	0.9776	0.8651	0.9776

6 Summary

The article proposes a model for detecting anomalies in Internet traffic. To this end a structure of an artificial neural network in the form of a variational autoencoder was used. The encoder module and groups of training data classes were used for classification. The experimental tests performed on authentic data collected from online stores proved the effectiveness of this method for detecting non-human website traffic. The results were compared with other popular methods used for anomaly detection. Putting the proposed solution into practice could help advertisers reduce financial losses incurred due to fraudulent activities and prevent unfair competition.

The problem under discussion is planned to be tested and compared with a classifier based on a feedforward neural network with a single output. However, it requires different data, where the pro-

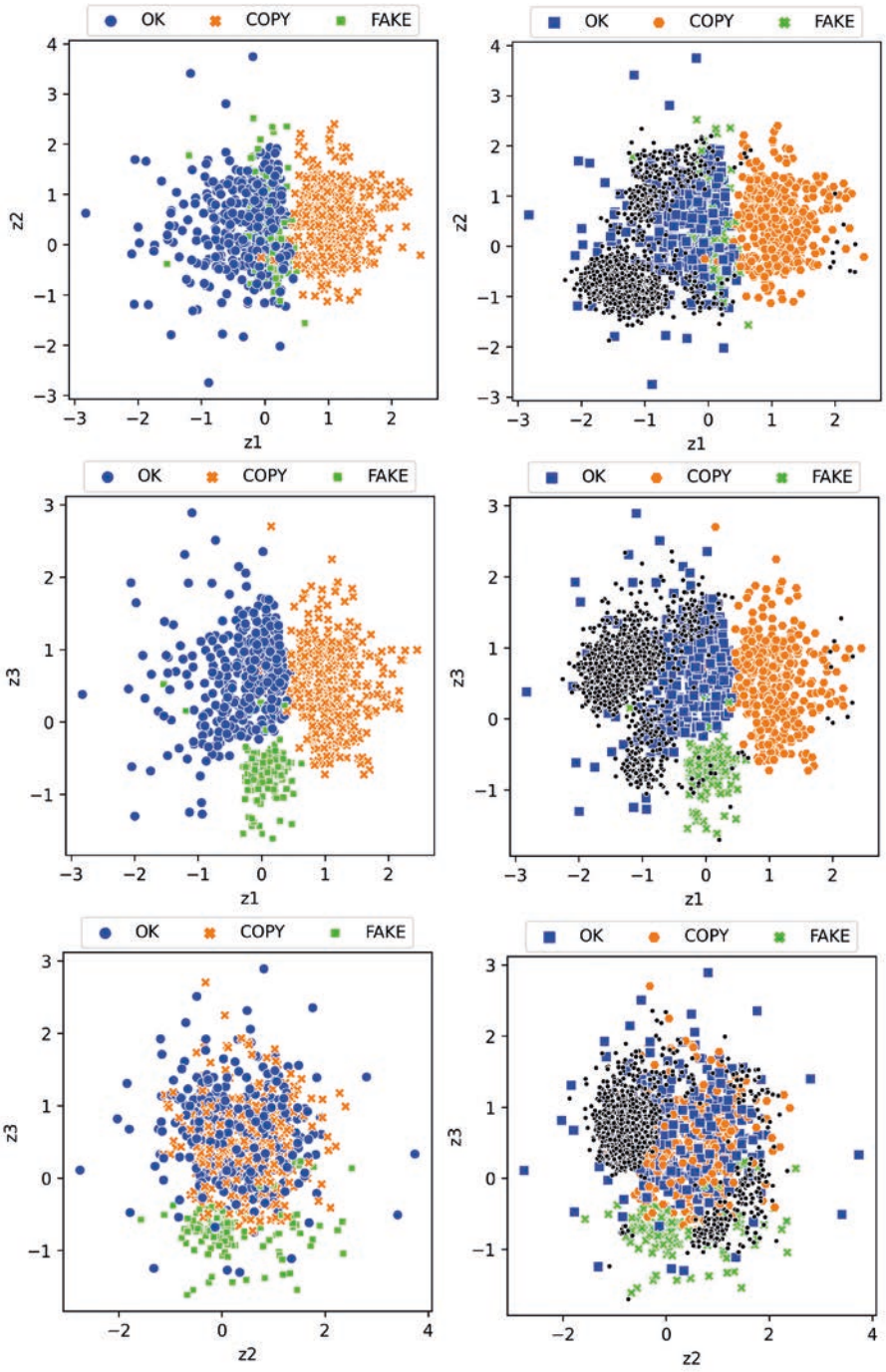


Figure 3. Visualization of latency space for different combinations of the dimensions z_i.

portion between correct and abnormal data will be more even. The algorithm can also be implemented using other mechanisms of computational intelligence, for example other artificial neural network structures [22], using parallel computing mechanisms or statistical methods.

References

[1] Recaptcha https://www.google.com/recaptcha/about/

[2] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. "Deep learning". MIT press, 2016.

[3] Zhou, Chong, and Randy C. Paffenroth. "Anomaly detection with robust deep autoencoders." Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, 2017.

[4] Farahnakian, Fahimeh, and Jukka Heikkonen. "A deep auto-encoder based approach for intrusion detection system." 2018 20th International Conference on Advanced Communication Technology (ICACT). IEEE, 2018.

[5] Q.P.Nguyen i in., GEE: A Gradient-based Explainable Variational Autoencoder for Network Anomaly Detection, 2019.

[6] https://www.emarketer.com/content/digital-ad-fraud-2019, 2019

[7] Barker S. ,"Future Digital Advertising, Artificial Intelligence & Advertising Fraud 2019-2023", Juniper Research, 2019

[8] Xiong, Yihui, and Renguang Zuo. "Recognition of geochemical anomalies using a deep autoencoder network." Computers & Geosciences 86 (2016): 75-82.

[9] Gabryel, Marcin, Konrad Grzanek, and Yoichi Hayashi. "Browser fingerprint coding methods increasing the effectiveness of user identification in the web traffic." Journal of Artificial Intelligence and Soft Computing Research 10 (2020).

[10] Gabryel, Marcin, et al. "Decision making support system for managing advertisers by ad fraud detection." Journal of Artificial Intelligence and Soft Computing Research 11 (2021).

[11] Kim, Taegong, and Cheong Hee Park. "Anomaly Pattern Detection in Streaming Data Based on the Transformation to Multiple Binary-Valued Data Streams." Journal of Artificial Intelligence and Soft Computing Research 12.1 (2022): 19-27.

[12] Brunner, Csaba, Andrea Kő, and Szabina Fodor. "An Autoencoder-Enhanced Stacking Neural Network Model for Increasing the Performance of Intrusion Detection." Journal of Artificial Intelligence and Soft Computing Research 12.2 (2022): 149-163.

[13] Santhosh, Kelathodi Kumaran, et al."Vehicular trajectory classification and traffic anomaly detection in videos using a hybrid CNN-VAE Architecture". IEEE Transactions on Intelligent Transportation Systems, 2021.

[14] Wang, Tian, et al. Generative neural networks for anomaly detection in crowded scenes. IEEE Transactions on Information Forensics and Security, 2018, 14.5: 1390-1399.

[15] Zhou, Yu, et al. VAE-based Deep SVDD for anomaly detection". Neurocomputing, 2021, 453: 131-140.

[16] An, Jinwon; Cho, Sungzoon. Variational autoencoder based anomaly detection using reconstruction probability". Special Lecture on IE, 2015, 2.1: 1-18.

[17] Pang, Guansong, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. "Deep Learning for Anomaly Detection: A Review." ACM Computing Surveys (CSUR) 54, no. 2 (2021): 1–38.

[18] Kingma, Diederik P.; Welling, Max. Auto-encoding variational Bayes". arXiv preprint arXiv:1312.6114, 2014.

[19] Tadeusz Inglot, Information Theory in the mathematical Statistics", Mathematica Applicanda", 42 (1), 2014, pp. 115–174

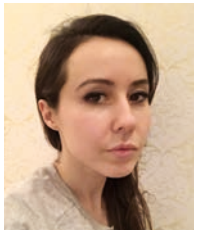
[20] Gabryel, Marcin, Lada, Dawid, Kocic, Milan "Autoencoder Neural Network for Detecting Non-human Web Traffic", ICAISC 2022, LNCS, Springer, accepted for printing.

[21] Zhao, Fangzhen, et al. "A Uniform Framework for Anomaly Detection in Deep Neural Networks." Neural Processing Letters (2022): 1-22.

[22] J. Bilski, B. Kowalczyk, A. Marjański, M. Gandor, J. Żurada, "A Novel Fast Feedforward Neural Networks Training Algorithm", Journal of Artificial Intelligence and Soft Computing Research, Vol.11, No. 4, 287-306 (2021), DOI: 10.2478/jaiscr-2021-0017

Marcin Gabryel earned his Ph.D degree in computer science at Czestochowa University of Technology, Poland, in 2007. He is an assistant professor in the Department of Computer Engineering at Częstochowa University of Technology. His research focuses on developing new methods in computational intelligence and data mining. He has published over 50 research papers. His present research interests include deep learning architectures and their applications in databases and security.
https://orcid.org/0000-0002-6701-0460

Dawid Lada graduated the Bechelor of Engineering degree in computer science at Częstochowa University of Technology in 2021 and now is a M.Sc. student in computer science in the Department of Computational Intelligence at Częstochowa University of Technology. His research focuses on the development of artificial neural networks in the sector of Internet traffic classification and anomaly detection with autoencoders.



Zofia Patora-Wysocka is a professor at the University of Social Science in Łódź, Poland. She received the Ph.D. degree from the Częstochowa University of Technology, Częstochowa, Poland in 2008, and the D.Sc. degree in economic sciences from the WSB University in Dąbrowa Górnicza in 2020. Her research interest includes change management, routine dynamics and strategy, practice theory, science and technology studies, and applications of data mining and artificial intelligence methods in management.
https://orcid.org/0000-0002-0429-0207

Zbigniew Filutowicz is an assistant professor at the University of Social Sciences in Łódź. He received the M.Sc. and Ph.D. degrees from Lodz University of Technology in 1973 and 1982, respectively. His research interests include human-computer communication, software engineering and applications of artificial intelligence in computer graphics and medical dialysis.
https://orcid.org/0000-0002-9543-9555

Marek Kisiel - Dorohinicki obtained Ph.D. in 2001 and D.Sc. in 2013 at the Department of Computer Science of the AGH University of Science and Technology in Krakow, Poland. His main research interests are metaheuristics, agent-based systems and criminal analysis. He works as a Full Professor in the Institute of Computer Science at the AGH University of Science and Technology.
https://orcid.org/0000-0002-8459-1877

Guang Yi Chen holds a B.Sc. in Applied Mathematics, an M.Sc. in Computing Mathematics, an M.Sc. in Computer Science, and a Ph.D. in Computer Science. During his graduate and postdoctoral studies in Canada, he was awarded many prestigious fellowships. He has published over sixty scientific journal papers in his fields and holds two granted US patents in image processing. He is currently affiliated to the Department of Computer Science and Software Engineering, Concordia University, Montreal, Quebec, Canada. His research interests include pattern recognition, image processing, machine learning, artificial intelligence, and scientific computing.
https://orcid.org/0000-0002-4811-2402