Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 176 (2020) 1015–1022

Procedia Computer Science

www.elsevier.com/locate/procedia

24th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

Influencing web customers. Deep learning practical application

Germanas Budnikas[a,b]* and Lukas Marculanas[a]

[a]*Applied Informatics Department, Informatics faculty, Kaunas University of Technology, Studentų g. 50, 51368 Kaunas, Lithuania*
[b]*Faculty of Economics and Informatics in Vilnius, Białystok University, Kalvarijų 135, 08221 Vilnius, Lithuania*

Abstract

With the global expansion of internet businesses, companies look for possibilities to increase their turnovers in virtual media. In these circumstances, it is significant for companies to employ online user behavior data in order to understand user intents and, to some extent, influence user online activity. This research examines the practical solution of the stated goal through an application of deep learning models. The contribution of the current study is an investigation of the success of real-time influence on web portal users to conduct specific activities leading to the achievement of predefined business goals. The study investigates several top-rated over-sampling techniques while solving class imbalance problem to evaluate their impact on the deep learning model performance. The study was conducted on the Lithuanian online educational platform. The obtained results indicate the feasibility of the suggested method.

Keywords: machine learning; recurrent neural networks; stratified cross-validation, user behavior; class imbalance problem.

1. Introduction

A vast majority of today's businesses is dependent on their activities on internet web sites and web services. Interaction between customers with online business platforms is achieved through the user interface (UI) elements used. Some goal-seeking customers could easily navigate and fulfill their needs. Another type of online visitors could be a bit disoriented or have no clear goal while browsing the website. At the same time, web site UI elements could

* Corresponding author. Tel.: +370 (37) 300 356.
E-mail address: germanas.budnikas@ktu.lt

10.1016/j.procs.2020.09.097

bring a visitor towards the attainment of an objective, the web site was designed for. The availability of recommendations guidance that could influence web customers to accomplish certain actions towards the fulfillment of some goals could be a solution in the described situation.

The goal could be achieved by employing deep learning methods to learn online customer behavior patterns. Obtained classifiers could process data about online visitor actions and give predictions about their further activity. These forecasts could be used in recommending the customers to activate specific web site UI elements, i.e., to accomplish a particular predefined business goal. A recommendation (or an influence) could be implemented by highlighting the appropriate UI element.

The approach presented in the paper consists of the following main phases: monitoring web site customer behavior, preparation of the customer online activity dataset for training using deep learning methods, creation and validation of the classifiers, coupling the best performed classifier with a web portal to predict online visitor intents and, use of the obtained forecasts for the influence.

The current investigation lies in the field of web personalization research area. This kind of the research includes Information Systems, Computer Science, Human-computer Interface and Marketing [1], [2]. Web personalization is an automated process including identifying a consumer, collecting consumer behavioral statistics, analyzing consumer preferences and fitting a content for each consumer [3]. Some recent works, similar to the suggested technique, will be outlined further, and similarities and differences will be discussed.

Paper [4] investigates web personalization issues in mobile shopping by analyzing motivations for m-shopping, which are rich content and functionality. Work [4] uses questionnaires to gather statistical data about an experience with m-shopping program that is popular in China. Investigation [5] utilizes diverse individualized information to improve recommendation accuracy. It proposes a hybrid recommendation model based on users' ratings, reviews, and social data. Whereas, the suggested in the paper approach collects statistics about accomplished activities, which were recommended to perform, on the fly and different analysis methods are used.

Paper [6] investigates the impact of e-service offerings in four online transaction phases (i.e., information, agreement, fulfillment, and after-sales stage) on customer purchasing intention using customer product transaction data from eBay. The results indicated that e-service offerings in the information phase are most influential on customers' willingness to pay. The following key differences are characteristic to [6]: influence on customer online behavior was investigated based on data about accomplished transactions, influence means were analyzed using only static information concerning the product; dynamic customer-oriented interaction based on customer intents was not considered in [6].

The effects of Facebook browsing and usage intensity on impulse purchase in f-commerce is investigated in [7]. This approach and currently presented one are similar in the analysis of impacts on the achievement of business targets in a web portal. However, [7] adopts questionnaire techniques to gather data and opinions from web portal users while the currently suggested approach does it in real-time.

The combined model of influencing online consumer behavior patterns for travel [8] used a nonlinear fuzzy network model to analyze synthetic data and questionnaires to evaluate respondent opinions. Whereas, the suggested in the paper approach employs real-time data to predict online customer intents, online customers were dynamically influenced by the results of the forecasts, and an efficiency of an influence was analyzed based on the outcomes of online visitor activities.

The approach proposed in the current paper is similar to [9] by the application of recurrent neural networks for a prediction of online customer real-time intents. Whereas the contribution of the current paper is in application of the obtained forecasts for a real-time influence on customer online activities and measuring the efficiency of such an impact.

The paper is structured in the following way. After the first introductory section, the second section gives a schema of the suggested approach and consists of sub-sections – the architecture of the web portal and used deep learning techniques. The third section presents a detailed description of the accomplished experiment. It is divided into two parts: 1) collecting the data and training the classifiers, and 2) influencing web visitors based on forecasts obtained from the prediction model. Paper ends with discussion and conclusions where key achievements and future research are outlined.

2. Schema of the suggested approach

The current work is further research of the authors on user behavior in web portals [10], [11]. The main result of the research lies in an insight that a series of online behavior may be interpreted as a set of preparation actions required to complete some predefined task that implicitly was introduced by web portal developers. This predefined task could be regarded as a target action or class. In this way, the activity of collecting users' on-site behavior can be seen as a process of gathering training data for supervised learning. Along with this, it is believed – machine learning methods will allow developing models for prediction of web portal user actions. Predictions obtained could be used as guidance to influence site visitor actions. The contribution of the current work is an investigation of the success of real-time influence on web portal users to conduct specific activities leading to the achievement of the target goal.

The following subsections briefly explain the main parts of the approach. First of all, this is discerning the main functional components and activities of the web portal that enable the accumulation of visitor statistics. Secondly – to address methods for performing a deep learning approach to execute the task under consideration. These methods are outlined in the subsection with a corresponding title.

2.1. Architecture of web portal

The proposed approach was tested by implementing it on an online education platform (web portal). The main web portal components (see Fig. 1.) fulfill specific business needs, e.g., they could be category management, post management, code example management. The visitor tracking component monitors customer activities. The visitor guidance component presents a web portal content and inspiration signs based on Prediction model forecasts. The inspiration signs (e.g., highlighting a button) are used as a means to encourage the visitor to perform a certain behavior on site. The behavior should lead to achieving a specific business target on a web portal. Clicking on certain UI components or visiting particular pages may correspond to business targets. A Prediction model forecasts the feasibility of the achievement of predefined business targets. The model's classifiers are trained based on statistical data of tracked visitor's behavior on a web site. In order to discover visitor's behavior patterns online, monitoring of specific visitor activities has to be performed. Current approach monitors the following web site visitor actions:

- Mouse clicking on specific buttons (like Signing up, Evaluating a post, Social share, Subscribe)
- Mouse clicking on ads
- Mouse moving
- Begin-, current-, end- mouse over an advertisement
- Opening a page

The monitored visitor actions enable the accumulation of a training dataset to be used in machine learning experiments in order to build classifiers. The classifiers are used to predict upcoming visitor action on a web site. An outcome from the best selected classifier is used to make a slight influence to a visitor by emphasizing a specific web site UI item. The UI item is associated with the business goal related to the visitor's current browsing activity. Preparation of a dataset and deep learning techniques used in the approach are outlined in the next subsection.

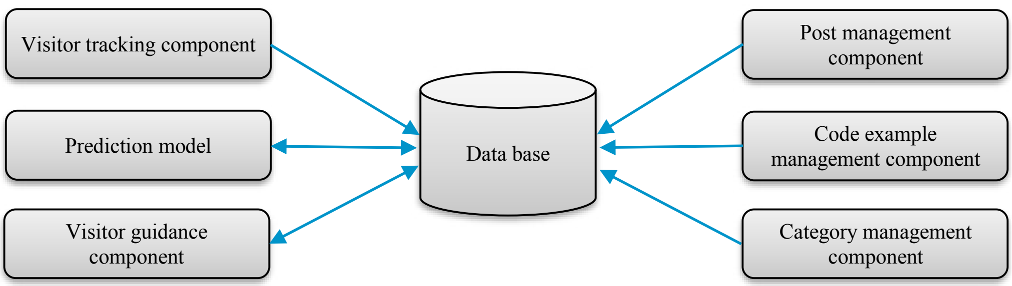

Fig. 1. Main architectural components used in the approach.

2.2. Deep learning techniques

The dataset that was composed under real-life conditions usually is characterized by imbalanced classes. A popular way to deal with imbalanced datasets is to either over-sample the minority class or under-sample the majority class. The suggested approach uses a popular Synthetic Minority Oversampling Technique (SMOTE) [12] and several its modifications. In this way it is possible to get a training set that is approximately class-balanced. The SMOTE instances are linear patterns of two similar instances from the minority class x and x' and are defined as $s = x + u \cdot x' - x$, where $0 \leqslant u \leqslant 1$ and x' is arbitrary selected among the 5 minority class akin neighbors of x.

The supervised learning approach can be applied to develop the classifier model because the data have clear classes that correspond to specific business goals to be achieved on a web portal. Different neural network architectures with feedback can be applied: Recurrent neural network (RNN), Long short-term memory (LSTM), or Gated recurrent unit (GRU).

Recurrent neural networks [13] are a class of artificial neural networks where neurons are connected to form a directed graph. Due to such interconnection of neurons, these networks have a short memory and are superior to neural networks without feedback in performing sequence-related tasks like analyzing web browsing behavior. Long-term memory is the architecture of a neural network with feedback that solves the problem of long-term dependency.

The key concept of LSTM [13] is a cell state. The state of the cell resembles a conveyor belt. It passes through the entire chain, undergoing minor linear transformations. The LSTM reduces or increases the amount of information in a cell state, depending on needs. To do this, carefully tuned structures called gates are used. These tuned structures can be: the input gate, forgetting gate, output gate, and a neuron cell.

- The input gate can allow or block the incoming impulse to change the state of the memory cell
- The forgetting gate controls the feedback and enable the cell to remember or forget a previous state
- The output gate allows or denies the state of the memory cell from affecting other neurons
- A neuron cell ensures that the state of the memory cell remains constant from one step to the next one when external interfaces are blocked

Gated recurrent units, first described in [14], are slightly more different than standard LSTMs. The forgetting gate and input gate are combined in gate recurrent unit and form one update gate. Besides, the state of the cell is combined with the hidden state, and there are other minor changes. The resulting model is more explicit than the standard LSTM.

The best-suited classifier for prediction usually is selected using cross-validation experiments. The resulting classifier is linked with a web portal. Details of this experiment are given in the next section.

3. Experiment: Lithuanian online education platform

The experiment was organized in the two phases: 1) collecting the data and training the classifiers; 2) influencing web visitors on the basis of forecasts obtained from a prediction model. Subsequent subsections provide the details.

3.1. Collecting the data and training the classifiers

The experiment was conducted on an online education platform, with approximately 500 thousand subscribers. It contains courses on IT-related fields in several languages. 2,352 million entries of visitor activities were collected during 28 days of an experiment in May 2019. Visitors were informed about a collection of statistics. The collected data were processed to prepare it for further analysis. The following actions were undertaken: a removal of empty entries, a grouping of browsing actions according to user sessions and web site business goals reached. The resulting table (see Tab. 1 below) gives an overview of the processed visitor browsing activities. The file for processing visitor history contained almost 123 thousand entries. The history data file was analyzed using standard deviation to remove entries-outliers in performed browsing activities. For instance, the highest number of accomplished actions by some visitors reached 2000, while its mean was 100. The sequence of activities performed by a visitor could include several target classes. The following 13 target classes were defined in the experiment: *none* – a visitor has not reached any goal; *code-try* – a visitor clicked on „Try it Live" button; *find coding exercises* – a visitor clicked on that button;

button-header-signup – a visitor clicked on „Sign up Free" button; *button-reserve-spot* – a visitor clicked on ad area in the bottom; *button-get-certified* – a visitor clicked on „Get Certified Now" button; *subscribe* – a visitor subscribed; *vote* – a visitor evaluated a post; *course-card* – a visitor clicked on an ad with advertised courses; *code-copy* – a visitor copied a code example; *side-banner* – a visitor clicked on ad area in the right; *button-get-started-now* – a visitor clicked on the „Get Started Now" button; *share* – a user clicked on one of the four social share buttons. In this way, the final number of entries was reduced to almost 122 thousand.

Table 1. A fragment of processed visitor browsing activities.

Achieved target class (business goal)	No. of visitor actions	Old or new visitor	Visitor browsing activities
none	1	new	JavaScript
none	4	new	HTML; mouse-move; mouse-move; scroll
code-try	11	new	HTML; mouse-move; scroll; mouse-move; scroll; HTML; mouse-move; best-code-editor; mouse-move; scroll

During the next step, a percentage of the occurrences of corresponding target classes was counted and presented in Tab. 2.

Table 2. A fragment of processed visitor browsing activities.

Achieved target class	Occurrence, %
none	87.5628
code-try	10.8796
find coding exercises	0.3176
code-copy	0.2872
button-reserve-spot	0.2044
button-header-signup	0.1699
subscribe	0.1641
vote	0.1403
course-card	0.0944
button-get-started-now	0.0722
button-get-certified	0.0575
side-banner	0.0419
share	0.0082

The data in the table show that almost 88% of the people that visited Web site, have not reached any goal. Another noticeable fact is that people come to the online educational platform site for code samples, as the other three classes most commonly achieved by users, involve live code testing, copying or code exercise search.

One more eye-catching issue is imbalanced classes in the collected dataset. The problem could be solved either by reducing data in the dataset or by synthesizing the new one. Besides the standard SMOTE procedure, three more techniques were employed to evaluate its impact on prediction model performance. They are a Combined Cleaning and Resampling (CCR) algorithm [15], Density-Based Synthetic Minority Over-Sampling Technique (DBSMOTE) [16], and another SMOTE extension – Iterative-Partitioning Filter (SMOTE-IPF) [17]. These three additional methods correspond to oversampling techniques with the highest Area Under the Curve (AUC) metrics according to [18] classifier rank when artificial neural networks are used for class predictions. A separation of training and test data was accomplished to verify a classifier with the original data (not the synthesized one) before applying the synthesizing techniques. When generating synthetic data, their amount increases significantly – approximately by 1 million entries. The synthetic data was used for the training of classifiers, more details about that – below.

Recurrent neural network models RNN, LSTM, and GRU were used to create classifiers using *Python* language library *hyperas*. The following parameters were investigated to obtain the right configuration: number of dropouts, optimization method, number of epochs, and batch size. A 5-fold stratified cross-validation technique was used to evaluate the performance of the three classifier models. Intel i7-6300k 4.6GHz with 16 GB RAM computer was used for computations, the appropriate results are introduced in Tab. 3.

Table 3. Results of the 5-fold stratified cross-validation experiments. Performance metrics – AUC.

Classifier model / Oversampling technique	LSTM	RNN	GRU
CCR [15]	*0.7412*	0.6917	*0.8361*
SMOTE-IPF [17]	0.7202	*0.6926*	0.8339
DBSMOTE [16]	0.7398	0.6817	0.8213
SMOTE [12]	0.7387	0.6903	0.8323

Tab. 3 shows an advantage of CCR combined cleaning and resampling algorithm when using LSTM and GRU. It complies with [18] investigation, which showed a benefit of CCR technique when a multi-layer perceptron model was used for different classification tasks. Tab. 3 shows the highest AUC metrics values when GRU classification model is used – it will be used further for a prediction.

3.2. Influencing web visitors based on forecasts obtained from the prediction model

GRU classifier was linked to the web portal to provide forecasts on visitor further activity based on a limited number of online actions performed. The study was conducted within 3 days. The prediction was carried out for the situations when the visitor performed 6 steps, and none of them belonged to any of the thirteen classes monitored. This number was chosen based on the average number of activities performed during the model training process.

When the required number of steps was collected (i.e., visitor performed 6 activities without reaching the target one), this sequence was sent to the prediction model that returned two target classes to which the visitor's actions were most likely. If the first, most likely to occur, class predicted was *none*, the second forecast was used instead. Based on the prediction, belonging to the class that is most likely to occur, a specific UI object was highlighted (see Fig. 2) to encourage a visitor to click on it. Afterward, it was monitored whether the highlighted item was activated. If the action actually performed was *none*, the second most probable forecast was used for influence.

A total of 10742 visitors visited the online education platform during the study period. The predictions were computed for 5638 visitors (see Tab. 4) and influence actions to visitor decisions were executed. 644 of them reached the target business goal.

The first three predicted target classes from Tab. 4 correspond to the three most used activities listed in Tab. 2. Following Tab. 4, the prediction model forecasted actions to occur based on 6 accomplished activities: *code-try* activity was forecasted 2410 times, *none* activity was forecasted 2350 times, etc. The forecasted activities were recommended to perform for corresponding visitors (see Tab. 4 second column). An exception was made for *none* activity – the second most likely to occur forecasted activity was proposed to perform. The 3rd column of Tab. 4 presents outcomes of influences undertaken. *code-try* action was performed 310 times at the 7th step.

Fig. 2. Illustration of visitor influencing (highlighting a button) based on the prediction model forecast.

Table 4. Prediction and influence statistics at 6^{th} – 7^{th} activities.

Predicted target class	First prediction counts after 6^{th} action	Influence statistics after 7^{th} action*
code-try	2410	310
none	2350	334
others	*insignificant*	

* Note that approximately 77% of visitors leaved the web portal after the 6^{th} action.

Table 5. Prediction and influence statistics at 6^{th} – 8^{th} activities for visitors who have not reached the target goal.

Predicted target class	Second prediction counts after 6^{th} action	Influence statistics after 8^{th} action
code-try	323	333
find coding exercises	11	1

As the resulted classes are highly imbalanced (see Tab. 5), the F1-score will be used for the evaluation of the influence.

Table 6. Summary of the influence performances.

	F1-score
Influence performance after the 7^{th} action	0.6499
Influence performance after the 8^{th} action	0.9848

As it was mentioned before, the prediction model returned two most likely predictions possible to occur given 6 actions that contained no target class. In case the 7^{th} action was *none*, the second most likely to occur prediction was used for the influence. Following this, 334 visitors did not reach any target class (their activity belonged to class *none*) during their 7^{th} action (see Tab. 4, third row). For such visitors, the second most likely to occur forecasted activities were proposed to complete. Tab. 5 outlines the outcomes of the prediction and influence processes after the 8^{th} action completed. Tab. 6 summarizes the performances of the influences at different browsing stages. Notably, the influence performance after the 8^{th} action is high. Reasons for such high value may include the following: either class *code-try* belongs to most often performed activities, or, visitors did not achieve any classes before and might be interested in clicking on the highlighted item, or a mixture of it.

4. Discussion and Conclusions

A further discussion about the feasibility of an approach to make an influence on web site visitor decisions to perform certain activities can take place. The main topic to examine is a cause that advanced a suggested behavior. Whether it was a genuine intent of a visitor to act, or it was exclusively caused by influence means or a mixture of causalities above – this still remains to be clarified.

The most commonly used measurement means (e.g., questionnaires) could be useful in this debate. However, questionnaires or similar measurement means were not desirable due to the politics of the commercial web portal used in the research. Influence metrics presented in the paper should be treated as a performance of a change that appeared due to the mixture of motives given formerly.

Investigation performed on the online education platform demonstrated a possibility to employ hidden data about visitor behavior to make predictions on future activities and, in turn, to influence their fulfillment. The following could be concluded based on the results of an experiment:

- The three most frequently predicted actions coincide with the three types of activities that are performed on the site most often. A performance of the influence is in linear dependency on 'popularity' of performed actions (see Tab. 2, Tab. 4, and Tab. 5)
- A performance of the influence increases with the number of empty activities (class *none*). The highest achieved F1-score value for the performance is 0.9848

Such an approach is useful for a business in several aspects:

- Possibility to recommend changes of certain UI items to improve their usability and 'popularity'
- Possibility to influence a visitor to achieve specific web portal business goals

Future works in the field of analysis of the behavior of web portal visitors include expanding the area of addressing this approach for different web portals, adjusting parameters of the method to increase the influence performance.

References

[1] Ville Salonen, and Heikki Karjaluoto (2016) "Web personalization: The state of the art and future avenues for research and practice" *Telematics and Informatics*, **33** (**4**): 1088-1104.

[2] Vinodh Krishnaraju, and Mathew K. Saji, and Vijayan Sugumaran. (2016) "Web personalization for user acceptance of technology: An empirical investigation of E-government services." *Information Systems Frontiers* **18**(**3**): 579-595.

[3] David Bodoff, and Shuk Ying Ho (2015) "Effectiveness of Website Personalization: Does the Presence of Personalized Recommendations Cannibalize Sampling of Other Items?" *International Journal of Electronic Commerce* **20**(**2**): 208-235.

[4] Jialiang Huang, and Liyun Zhoub (2018) "Timing of web personalization in mobile shopping: A perspective from Uses and Gratifications Theory" *Computers in Human Behavior* **88**:103-113.

[5] Ji, Zhenyan and Pi, Huaiyu and Wei, Wei and Xiong, Bo and Woźniak, Marcin and Damasevicius, Robertas. (2019). "Recommendation Based on Review Texts and Social Communities: A Hybrid Model" *IEEE Access* **7**: 40416-40427.

[6] Xun Xu and Shuo Zeng and Yuanjie He. (2017) "The influence of e-services on customer online purchasing behavior toward remanufactured products" *International Journal of Production Economics* **187**: 113–125.

[7] Lai-Ying Leongab, and Noor Ismawati Jaafar, and Sulaiman Aininc (2018) "The effects of Facebook browsing and usage intensity on impulse purchase in f-commerce", *Computers in Human Behavior* **78**: 160-173.

[8] Wen-Bao Lin and Ming-Kuen Wang and Kevin P. Hwang. (2010) "The combined model of influencing online consumer behavior" *Expert Systems with Applications* **37**(**4**): 3236–3247.

[9] Sakar, C. Okan, and Polat, S., and Katircioglu Mete, and Kastro Yomi. (2018) "Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks." *Neural Computing and Applications* **31**: 6893–6908.

[10] Germanas, Budnikas (2017) "A Model of Machine Learning Based Interactive E-business Website." *In: Saeed K., Homenda W., Chaki R. (eds) Computer Information Systems and Industrial Management. CISIM 2017. Lecture Notes in Computer Science* **10244**: 470–480.

[11] Germanas, Budnikas (2015) "A Model for an Aggression Discovery Through Person Online Behavior." *In: Saeed K., Homenda W. (eds) Computer Information Systems and Industrial Management. CISIM 2015. Lecture Notes in Computer Science* **9339**: 305–315.

[12] N.V. Chawla, and K.W. Bowyer, and L.O. Hall, and W.P. Kegelmeyer (2002) "SMOTE: synthetic minority over-sampling technique" J. Artificial Intelligence Res. **16:** 321–357.

[13] Alex Graves (2012) "Supervised Sequence Labelling with Recurrent Neural Networks." *Springer*

[14] Cho, K., van Merrienboer, and B., Gulcehre, and C., Bougares, and F., Schwenk, H., and Bengio, Y. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *In Conf. on Empirical Methods in Natural Language Processing* 1724–1734.

[15] Michał Koziarski and Michał Wozniak (2017) "CCR: A combined cleaning and resampling algorithm for imbalanced data classification." *Int. J. Appl. Math. Comput. Sci.* **27**: 727–736.

[16] DBSMOTE: C. Bunkhumpornpat, and K. Sinapiromsaran, and C. Lursinsap (2012) "DBSMOTE: Density based synthetic minority over-sampling technique", *Appl. Intell.* **36** (3): 664–684.

[17] J.A. Sáez, and J. Luengo, and J. Stefanowski, and F. Herrera (2015) "SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering." *Inf. Sci.* **291**: 184–203.

[18] György Kovács. (2019) "An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets." *Applied Soft Computing* **83**: 105662.