2019 6th International Conference on Control, Decision and
Information Technologies (CoDIT'19) | Paris, France / April 23-26, 2019

Deep Neural Network Approach for Predicting the Productivity of Garment Employees

Abdullah Al Imran*, Md Nur Amin†, Md Rifatul Islam Rifat‡, Shamprikta Mehreen§
*,†Dept. of Computer Science & Engineering
American International University-Bangladesh, Dhaka, Bangladesh
‡Dept. of Electronics and Telecommunication Engineering
Rajshahi University of Engineering & Technology, Rajshahi, Bangladesh
§Dept. of Computer Science
University Nice-Sophia Antipolis, Nice, France
Email: {*abdalimran, †nuramin.aiub, ‡irifat.ruet, §shamprikta2244}@gmail.com

***Abstract*—The garment industry is one of the most dominating industries in this era of industrial globalization. It is a highly labor-intensive industry that requires a large number of human resources to produce its goods and fill up the global demand for garment products. Because of the dependency on human labor, the production of a garment company comprehensively relies on the productivity of the employees who are working in different departments of the company. A common problem in this industry is that the actual productivity of the garment employees sometimes does not meet the targeted productivity that was set for them by the authorities to meet the production goals in due time. When the productivity gap occurs, the company faces a huge loss in production. This study aims to solve this problem by predicting the actual productivity of the employees. To achieve this aim, a Deep Neural Network (DNN) model has been proposed to predict the actual productivity of the employees. The experimental results of this study have shown that the proposed model yields a promising prediction performance with a minimal Mean Absolute Error (=0.086) which is less than the baseline performance error (=0.15). Such prediction performance can indisputably help the manufacturers to set an accurate target, minimize the production loss and maximize the profit.**

***Index Terms*—Deep Neural Networks, Regression, Garments Manufacturing, Productivity Prediction.**

I. Introduction

The apparel industry forms a major part of manufacturing production, employment, and trade in many developing countries. It plays a key role in the growth of the economy and development of a country as it produces a massive amount of foreign currency. For instance, Bangladesh is a developing country which is the second largest apparel exporting country in the world [1]. According to the Export Promotion Bureau (EPB) data that have recently released, Bangladesh's export earnings from the Ready-Made Garments (RMG) sector stood at $30.61 billion and RMG holds almost 14.07% of the GDP of Bangladesh as well as the 81% of the total export earnings [2]. Apart from the economic perspective, the garment industry is one of the most labor-intensive industries in the world. About 75 million people are directly involved in the textile, clothing, and footwear sector worldwide [3] and therefore the production of the garments industry comprehensively depends on the competency and productivity of the employees. That is why industry entrepreneurs must have a strategic leadership vision to enhance the production and keep pace with the sudden changes in the developing sector [4]. Gunawan et al. [5] have identified that the enhancement of the managerial capabilities in financial monitoring and evaluating the activities in Indonesian Garment Medium Scale Manufacturers (IGMS) is indispensable to sustain Indonesia's economy.

Over the past few decades, a considerable amount of studies has been published in the discipline of the garments industry that is concerned with the enhancement of the production, improvement of the quality and minimization of the manufacturing cost, and reduction of the production complexity. However, there exist a few studies in this discipline that have also concerned with the application of data mining and machine learning techniques in this sector. Intuitively, data mining and machine learning is a process that is used to discover the concealed valuable knowledge by analyzing a massive amount of data [6]. The garments industry produces a vast amount of data from the daily labor intensive activity and industrial process. This data can be utilized using advanced computational techniques to extract the hidden insights and enhance their production performance as well as solving many of their industrial and managerial problems.

Covering the largest number of people with the fewest number of sizes is the optimal scenario in the context of tailoring. Therefore, through the use of objective interestingness measures-based feature selection and feature extraction, Viktor et al. [7] have discovered the relevant subsets of body measurements. Likewise, Hsu et al. [8] have applied comprehensive factor analysis and a decision tree-based data mining approach to develop an innovative sizing system for male workers in Taiwan to facilitate the manufacturing process. Usually, people seek for the products of high quality but in low cost and thus manufacturing the products without any defect turned into a challenge to the manufacturers so as to fulfill the demand of the customers. Lee et al. [9] presents a hybrid Online Analytical Processing (OLAP)-association rule mining based intelligent quality management system so that manufacturers are able to explore the garment defect patterns

978-1-7281-0521-5/19/$31.00 ©2019 European Union

in a timely manner to improve the quality. To achieve better quality assurance in the garment industry, Hsu et al. [10] have developed a slippery genetic algorithm-based process mining system (sGAPMS) to optimize the fuzzy association rule. Rahim et al. [11] have proposed a methodology for mining the industrial engineered manufacturing data of garment industry. Based on the two-stage cluster approach namely, Ward's minimum variance method and K-means algorithm, Hsu et al. [12] have developed a data mining framework to extract the valuable patterns and improve the industrial standards so as to enhance production. Like the aforementioned study [10], Lee et al. [13] have focused on the quality assurance in the garment industry and the eminent authors of this paper deployed a radio frequency identification (RFID) based recursive process mining system. This proposed recursive process mining algorithm can extract the relationships between process parameters and product quality that can be exploited for quality assurance.

From the above discussion, we can understand that some previous studies have employed data mining techniques in the garment industry. Most of the studies focused on proposing measurement and sizing systems, quality management systems, frameworks for pattern extraction and so on. However, no studies have been found employing advanced computational techniques for predictive modeling to forecast manufacturing productivity. In this study, we have aimed to predict the productivity of the garment employees using an advanced computational technique. We have proposed a Deep Neural Network (DNN) based approach for predicting the performance of the employees in the garment industry. We have conducted an experiment by applying our proposed model to the production data of a reputed Bangladeshi garment manufacturing company. The experimental results of this study have shown that our proposed model can predict the actual productivity of the employees very closely and significantly improve the prediction performance with less Mean Absolute Error (=0.086) than the baseline performance score (=0.15). Therefore, we believe this study will undoubtedly help the garment manufacturing authorities to solve such industrial problems and take more reliable data-driven strategies to develop the production process.

II. Problem Background

A complete garment production pipeline consists of several sequential processes such as designing, sample confirmation, sourcing and merchandising, lay planning, marker planning, spreading and cutting, sewing, washing, finishing, and packaging. Along with maintaining this pipeline, a successful garment production always follows a line plan that defines when the production will be started, how many pieces are expected and when the order needs to be completed. To complete a whole production within a bounded time, these sequential processes need to be to performed efficiently in a timely manner. In order to meet the production goals, the associated industrial engineers strategically set a targeted productivity value against each working team in the manufacturing process. However, it is a common scenario that the actual productivity of the working teams does not always meet their targeted productivity because of various factors such as the number of employees in a team, the amount of work in progress, over-time, money incentive and so on. This gap between the targeted and actual productivity of the working teams often hampers the production to be completed in due time. This problem causes a huge financial loss for a garment manufacturing company. Therefore, while developing a line plan, an industrial engineer should set the targeted productivity in a precise and data-driven manner so that the gap between the targeted and actual productivity is minimized. If they can predict the actual productivity of the working teams and set the target according to the prediction, then they can certainly minimize the financial loss, maximize the profit and increase the production efficiency.

This study aims to solve the problem with a data-driven advanced computational technology called Deep Neural Networks (DNN). In this study, we have proposed a DNN model for predicting the productivity of the garment employees based on their previous data.

III. Description of Dataset

In this study, the dataset we have collected was originated from the Industrial Engineering (IE) department of a garment manufacturing unit of a reputed company situated in Bangladesh. The collected dataset contains the production data of the sewing and finishing department for three months from January 2015 to March 2015 of the company. The dataset consists of 1197 instances and includes 13 attributes. Table I provides the data dictionary of the dataset.

TABLE I
Data Dictionary

Attribute	Description
date	Date in MM-DD-YYYY
department	Associated department with the instance
team_no	Associated team number with the instance
no_of_workers	Number of workers in each team
no_of_style_change	Number of changes in the style of a particular product
targeted_productivity	Targeted productivity set by the authority for each team for each day
smv	Standard Minute Value, it is the allocated time for a task
wip	Work in progress. Includes the number of unfinished items for products
over_time	Represents the amount of overtime by each team in minutes
incentive	Represents the amount of financial incentive (in BDT) that enables or motivates a particular course of action
idle_time	The amount of time when the production was interrupted due to several reasons
idle_men	The number of workers who were idle due to production interruption
actual_productivity	The actual productivity value which ranges from 0.0 to 1.0

IV. Data Preprocessing

The most essential phase of solving a real-world problem using machine learning is to preprocess the dataset properly. It is one of the primary key factors for obtaining a promising result from the model. In real world context, data can be found noisy, malformed, incomplete, missing and imbalanced. Before fitting the data into a machine learning model, it is necessary to make the data suitable for the model so that it can learn the underlying patterns in the data precisely. In a machine learning pipeline, the data preprocessing phase may include a bunch of tasks according to the form and structure of the data such as data cleaning, feature encoding, instance selection, normalization, transformation, feature extraction and selection. In this study, we have structured this phase according to the form of our dataset which includes feature engineering, feature encoding, feature scaling, and data partitioning. All the tasks are discussed briefly in the subsections below.

A. Feature Engineering

Feature engineering is the act of extracting features from raw data and transforming them into formats that are suitable for the machine learning model [14]. In our dataset, we have a feature named "date" containing temporal information; is needed to be processed in order to feed the model. We performed feature engineering techniques to extract five distinct features from it. The five features are "year", "month", "quarter", "week" and "day". After performing feature engineering our dataset contained a total number of 16 input features.

B. Feature Encoding

Feature encoding is the process of representing categorical variables into numeric form. As the neural networks can only learn from numeric data, it is mandatory to encode the categorical data into numbers before feeding the data to the model. In our dataset, we have 5 categorical features namely "department", "team_no", "month", "quarter", and "year". We have applied the One-hot encoding technique to encode these variables into a suitable numeric form. By definition, One-hot encoding or dummy variables indicate categories or subgroups using a 0 (absent) or 1 (present) value [15]. This creates a binary column for each category and returns a sparse matrix or dense array. Basically, a one-hot encoder derives the categories based on the unique values in each feature. After applying the one-hot encoding technique to the categorical features, we have finally got a total number of 30 input features to feed the model.

C. Feature Scaling

Feature scaling is a data standardization process which scales the data to a fixed range and ends up with smaller standard deviations, which suppresses the effect of outliers. It is a very essential part in terms of training a neural network based model as it helps the neural network weights converge faster. As we are using the neural network model, it is our primary need to scale the features to boost its learning performance. For feature scaling, we have used the MinMaxScaler [16] from scikit-learn which uses the following formula:

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

D. Data Partitioning

In this study, we have applied the training-validation-testing approach. As we are training a deep neural network model, we need to tune the hyperparameters and be cautious about model overfitting or underfitting. Hence, a validation set will help us in tuning the hyperparameters and assessing if the model is learning properly or not. From the total number of 1197 instances, we have used 80% (=969) instances for training, 10% (=108) for validation, and 10% (=120) for testing the model.

V. Methodologies

In this study, we have maintained a proper workflow for our experiment. The workflow starts with designing the architecture of the DNN model followed by the selection of the evaluation metrics and defining a baseline performance score for the model. All the implementations in this study have been performed using the Python programing language, scikit-learn [16], and Keras [17].

A. The Architecture of the DNN Model

Designing an optimal neural network architecture requires consideration of a few high-level parameters. The parameters could be the number of hidden layers, the number of neurons in the input, output, and hidden layers, the activation functions, choosing a good optimization algorithm, defining the learning rate, the optimal number of epochs, batch size, and few others. All of these parameters are used combinedly to design a neural network and they have an enormous impact on the learning process of the model and its ultimate performance. These are the parameters that cannot be trained and need to be fine-tuned or selected with experience and judgment. Fig. 1 represents the visual architecture of our proposed model.

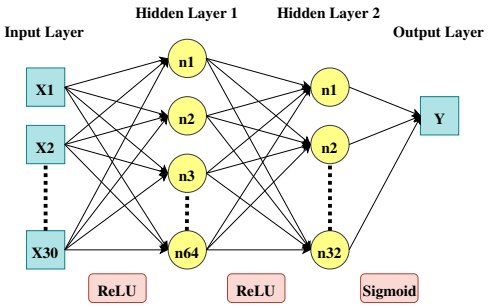

Fig. 1. DNN Architecture.

1) The Number of Neurons in the Input and Output Layers: A neural network architecture starts with specifying the number of neurons in the input and output layers. Determining the number of neurons in the input layer depends on the shape of the training data. Our training data has a shape of (969, 30), which means there are 969 instances and 30 input features. According to the neural network terminology, we specify 30 neurons in the input layer as there are 30 input features in the training data. As the problem is a regression problem and our expected output is a single numeric value, the output layer contains only one neuron.

2) The Number of Hidden Layers and Neurons: Determining the number of hidden layers and neurons is a major factor for designing the structure of a neural network as it defines the complexity and efficiency of the network. A study by Jeff Heaton [18] explains that, networks with a single hidden layer can approximate any function that contains a continuous mapping from one finite space to another, networks with two hidden layers can represent an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy, and networks with more than three hidden layers can learn complex representations and perform feature engineering automatically. So, according to the ideas of Jeff, designing the model with 2 hidden layers is a good fit for our problem. As our training data is not very large, it is unnecessary to take more than two hidden layers since it will make the model very complex, computationally inefficient and may cause overfitting to the training data.

We have used a rule of thumb from [19] to determine the number of neurons in the hidden layers. The book suggested that the number of hidden neurons in the first hidden layer should refer to the number of input dimensions. If the final number of input dimension in a given training dataset is x, we should use at least the closest number to $2x$ in the power of 2. As our final input dimension is 30, we have chosen 64 neurons for the first hidden layer since it is the closest number to $2x$ in the power of 2. Since the second hidden layer is the last hidden layer before the output, we have chosen 32 neurons which the closest number to $\frac{2x}{2}$ in the power of 2.

3) Activation Functions: Activation functions perform the functional mappings between the inputs and response variable. It basically converts an input signal of a node into an output signal. That output signal is then used as an input in the next layer in the stack. The main role of activation functions is to make the networks non-linear. In this study, we have used the following two activation functions.

Rectified Linear Unit (ReLU): ReLU [20] is a most commonly used activation function in deep learning. The function returns 0 if it receives any negative input, but for any positive value x, it returns that value back. It can be written as-

$$f(x) = max(0, x) \tag{2}$$

where x is the input to a neuron. ReLU allows the model to account for non-linearities and interactions so well. In the first and second hidden layers, we have used the ReLU activation function.

Sigmoid: A sigmoid [21] function produces a curve with an 'S' shape and ranges from 0 to 1. It is often used in neural networks to introduce nonlinearity in the model. A neural network element computes a linear combination of its input signals and applies a sigmoid function to the result. As our desired output ranges from 0 to 1, we have selected this function in the output layer. A sigmoid function is written as,

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

4) Optimization Algorithm: In neural networks, an optimization algorithm plays the most important role in achieving a promising result from the model. It shapes the model into its most accurate possible form by adjusting the weights. There are many optimization algorithms that are used in neural networks such as Gradient Descent, Adagrad, RMSprop, Adam and so on. However, we have used the Adam optimization algorithm which is described below.

Adaptive Moment Estimation (Adam): Adam [22] is one of those algorithms that work well across a wide range of deep learning architectures. It is a combination of gradient descent with momentum and RMSprop algorithms. The primary advantage of using Adam is that it has relatively low memory requirements and usually works well even with a little tuning of hyperparameters. The Adam algorithm can be formulated as follows:

$For\, each\, parameter\, W_j$:

$$v_t = \beta_1 \times v_{t-1} - (1 - \beta_1) \times g_t \tag{4}$$

$$s_t = \beta_2 \times s_{t-1} - (1 - \beta_2) \times {g_t}^2 \tag{5}$$

$$\Delta w_t = -\eta \frac{v_t}{\sqrt{s_t + \epsilon}} \times g_t \tag{6}$$

$$w_{t+1} = w_t + \Delta w_t \tag{7}$$

where,
$\eta =$ Initial learning rate
$g_t =$ Gradient at time t along w_j
$v_t =$ Exponential average of gradients along w_j
$s_t =$ Exponential average of squares of gradients along w_j
$\beta_1, \beta_2 =$ Hyperparameters

5) Evaluation Metrics: To evaluate our model performance and measure the errors, we have considered three evaluation metrics named Mean Squared Error, Mean Absolute Error, and Mean Absolute Percentage Error. All the metrics have been chosen by considering their benefits and interpretation. Details about the metrics are discussed below.

Mean Squared Error (MSE): MSE basically measures the average squared error of our predictions. For each point, it calculates the square difference between the predictions and the target and then averages those values. The higher this value, the worse the model is. MSE can be formulated as:

$$\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{8}$$

Mean Absolute Error (MAE): MAE is calculated as an average of absolute differences between the target values and the predictions. The MAE is a linear score which means that all the individual differences are weighted equally in the average. What is important about this metric is that it penalizes huge errors that not as that badly as MSE does. Thus, it is not that sensitive to outliers as mean square error. MAE can be formulated as:

$$\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{9}$$

Mean Absolute Percentage Error (MAPE): MAPE is basically expressed as the relative error preference. For each instance, the absolute error is divided by the target value, giving a relative error. The primary advantage of using MAPE is the clear interpretability of its results. MAPE provides a single value of percentage for the error. Therefore, when the average range of the prediction is known, it can be simply estimated that what the predictions are going to look like. MAPE can be formulated as:

$$\frac{100\%}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i} \tag{10}$$

6) Defining Baseline Performance: To define a baseline performance for a better assessment of our model, we have estimated the mean value of the target variable in the training dataset and roughly considered it as the prediction for all the test samples. Then we have calculated the mean absolute error to determine the baseline value. The calculation has been performed as follows:

$$\bar{X} = \frac{1}{N_{tr}}\sum_{i=1}^{N_{tr}} y\_train_i \tag{11}$$

$$baseline\ score = \frac{1}{N_{te}}\sum_{i=1}^{N_{te}}\left|(y\_test_i - \bar{X})\right| \tag{12}$$

where,
$\bar{X}$ = mean of training outcomes
N_{tr} = number of instances in the training data
y_train = target variable in the training data
N_{te} = number of instances in the test data
y_test = target variable in the test data

The baseline performance we have got after the calculation is 0.146. To obtain a promising result, our model should deliver a performance score less than the baseline score.

VI. Result and Analysis

The results of the performance of our proposed model have been measured using the aforementioned three evaluation metrics namely MSE, MAE, and MAPE. The numeric values of the results has been recorded upto 3 decimal places. The analysis of the models' performance has been performed in two steps. Firstly, we have produced a learning curve and analyzed the training and validation performance over time which helped us to diagnose if there is any underfitting or overfitting. Secondly, we have used the final trained model to predict on the test dataset which helped us to analyze how the model will perform in real world.

To produce the learning curve, we have plotted the MAE resulted from the loss function and trained the model up to 50 epochs. Fig. 2 represents the losses with respect to the epochs.

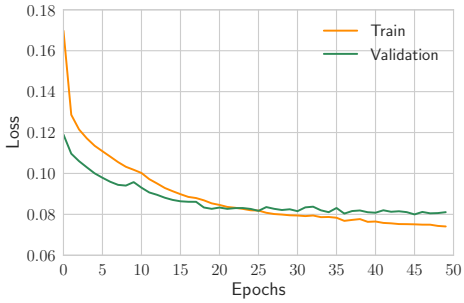

Fig. 2. Train vs Validaion loss across epochs.

From Fig. 2, we can observe that the training and validation losses got converged at 25^{th} epochs. The validation loss has been quite consistent in converging with training loss. This is the sign of a well-fitted model.

After the training and validation phase, we have used the model to predict on the test data. The test data was containing 120 instances. The obtained results from training, validation, and testing are presented in Table 2.

TABLE II
Performance of the Model

	MSE	MAE	MAPE
Training	0.020	0.091	16.883
Validation	0.018	0.089	15.949
Test	0.018	0.086	15.932

From Table 2, it can be observed that our obtained test results are very close to the validation results, which means that the model has learned the underlying patterns well from the data without overfitting. Moreover, we can see that in all the phases the results are quite consistent with very minimal variation.

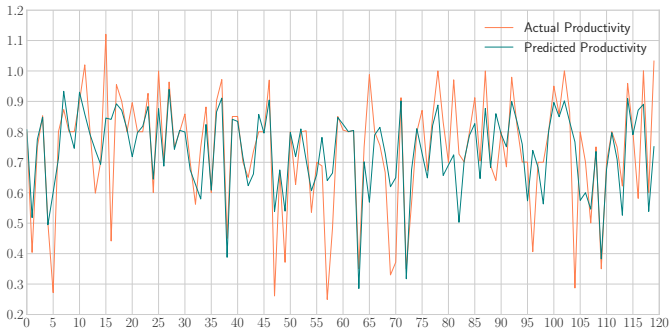

Fig. 3. Actual Productivity vs Predicted Productivity.

To understand the variation in actual and predicted productivity more clearly, we have represented all the 120 test cases into a line graph in Fig. 3. From the line graph, we can clearly observe that except few points, all the predicted values have very less variation with the actual value. For most of the points, the model learned very well and could predict the actual value precisely.

All the above analysis and error metrics shows that our model has learned the underlying pattern from the data very well and can promisingly predict the actual productivity of the garment employees.

VII. Conclusion

In this study, our aim was to propose a Deep Neural Network (DNN) based model for predicting the productivity performance of the garment employees. The dataset we have used is a real-world dataset collected from a renowned garment manufacturing company in Bangladesh. We performed a bunch of data preprocessing tasks such as feature engineering, scaling, and encoding. For designing our proposed DNN model, we have followed some proven methodologies from few recognized research studies. In the DNN architecture, we have selected the hidden layers, neurons, optimization algorithm and tuned the hyperparameters by manual search and practical judgment. The performance of the model was evaluated based on 3 evaluation metrics named MSE, MAE, and MAPE. For examining the training and validation loss we have used MAE as the loss metric. The training and validation performance of the model showed a consistent learning curve with very less deviation. The testing performance of the model was also very promising in terms of all the 3 metrics. As test performance, the model yielded a value of 0.018 for MSE, 0.086 for MAE, and 15.932 for MAPE. The test MAE (=0.086) performance was significantly better than the baseline MAE (=0.15). Most of the predicted values on the test data were very close to the actual values and overall there was very less variation. The performance of our proposed model proved its consistency in learning from the productivity data of garment employees and shown a promising prediction performance.

References

[1] Mirdha, R. U. (August 11, 2017). Exporters hardly grab orders diverted from China. The Daily Star, Retrieved from https://www.thedailystar.net/business/exporters-hardly-grab-orders-diverted-china-1446907, Accessed: 4 January, 2019

[2] Islam, M. S., Rakib, M. A., & Adnan, A. T. M. (2016). Ready-Made Garments Sector of Bangladesh: Its Contribution and Challenges towards Development. Stud, 5(2).

[3] 'Global Fashion Industry Statistics - International Apparel' (Fashion United), Retrieved from http://www.fashionunited.com/global-fashionindustry-statistics-international-apparel, Accessed: 4 January, 2019

[4] Nimlaor, C., Trimetsoontorn, J., & Fongsuwan, W. (2015). Thai garment industry organisational characteristics and strategic factors that affect corporate performance. International Journal of Management and Enterprise Development, 14(2), 126-143.

[5] Gunawan, A., Wahdan, M., & Van Den Herik, H. J. (2010). Increasing the managerial capabilities in Indonesian garment manufacturing. International Journal of Economic Policy in Emerging Economies, 3(4), 346.

[6] Hand, D. J. (2007). Principles of data mining. Drug safety, 30(7), 621-622.

[7] Viktor, H. L., Peña, I., & Paquet, E. (2012). Who are our clients: consumer segmentation through explorative data mining. International Journal of Data Mining, Modelling and Management, 4(3), 286-308.

[8] Hsu, C. H., & Wang, M. J. J. (2005). Using innovative technology to establish sizing systems. International journal of innovation and learning, 2(3), 233-245.

[9] Lee, C. K. H., Choy, K. L., Ho, G. T., Chin, K. S., Law, K. M. Y., & Tse, Y. K. (2013). A hybrid OLAP-association rule mining based quality management system for extracting defect patterns in the garment industry. Expert Systems with Applications, 40(7), 2435-2446.

[10] Lee, C. K. H., Choy, K. L., Ho, G. T., & Lam, C. H. (2016). A slippery genetic algorithm-based process mining system for achieving better quality assurance in the garment industry. Expert systems with applications, 46, 236-248.

[11] Rahim, M. S., Rahman, M., & Chowdhury, A. E. (2017). Mining Industrial Engineered Data of Apparel Industry: A Proposed Methodology. International Journal of Computer Applications, 161(7).

[12] Hsu, C. H. (2009). Data mining to improve industrial standards and enhance production and marketing: An empirical study in apparel industry. Expert Systems with Applications, 36(3), 4185-4191.

[13] Lee, C. K. H., Ho, G. T. S., Choy, K. L., & Pang, G. K. H. (2014). A RFID-based recursive process mining system for quality assurance in the garment industry. International journal of production research, 52(14), 4216-4238.

[14] Zheng, A., & Casari, A. (2018). Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists." O'Reilly Media, Inc.".

[15] Marko. Mooi Sarstedt (Erik.). (2018). Concise Guide To Market Research: The Process, Data, And Methods Using Ibm Spss Statistics. Springer-verlag Berlin An.

[16] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

[17] Chollet, F. (2015). Keras.

[18] Heaton, J. (2008). Introduction to neural networks with Java. Heaton Research, Inc.

[19] Moolayil, J. (2019). Learn Keras for Deep Neural Networks. Apress. https://doi.org/10.1007/978-1-4842-4240-7

[20] Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., & Seung, H. S. (2000). Erratum: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. Nature, 405(6789), 947–951. https://doi.org/10.1038/35016072

[21] Han, J., & Moraga, C. (1995, June). The influence of the sigmoid function parameters on the speed of backpropagation learning. In International Workshop on Artificial Neural Networks (pp. 195-201). Springer, Berlin, Heidelberg.

[22] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.