Sensors and Materials, Vol. 35, No. 10 (2023) 4671–4680
MYU Tokyo

S & M 3421

Training a Neural Network to Predict House Rents Using Artificial Intelligence and Deep Learning

Yonghu Yang,[1] Hong-Mei Dai,[2*] Chung-Hsing Chao,[3]
Sufen Wei,[4] and Cheng-Fu Yang[5,6**]

[1]School of Artifical Intelligence, Dongguan City College, Guangdong, 523419, P.R. China
[2]College of Art, Zhongqiao, Shanghai Zhongqiao Vocational and Technical University, Shanghai 201514, China
[3]Department of Intelligent Vehicles and Energy, Minth University of Science and Technology, Hsinchu, Taiwan
[4]School of Ocean Information Engineering, Jimei University, Xiamen 361021, China
[5]Department of Chemical and Materials Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan
[6]Department of Aeronautical Engineering, Chaoyang University of Technology, Taichung 413, Taiwan

(Received July 15, 2023; accepted September 26, 2023)

Keywords: neural network, artificial intelligence, deep learning, rental housing price

Our focus lies on the advancement of predictive capabilities through the training of neural networks, employing the remarkable technologies of artificial intelligence and deep learning, which are being used in the dynamic realms of the real estate and financial sectors. Rent forecasting is emerging as a pivotal application scenario and presents immense potential for real estate agents, financial institutions, and property developers alike. This powerful tool provides them with the ability to make informed decisions in a rapidly changing market environment. Today, it is not uncommon to see numerous real estate websites harnessing the potential of machine learning models to offer rental price predictions, thereby simplifying the decision-making process for both tenants and landlords. The integration of machine learning models has also become increasingly prevalent among forward-thinking real estate firms and financial institutions, leading to more precise rent determinations. However, it is important to acknowledge that the endeavor of training neural networks for rent prediction remains a relatively nascent field. Continued research and experimentation are vital in our pursuit of the improvement of both the performance and accuracy of these models. As we navigate this exciting frontier, we anticipate significant advancements that will reshape the landscape of rent prediction within the real estate and financial industries.

1. Introduction

Rent market transparency can be enhanced through the utilization of neural networks to forecast rental prices.[1,2] By employing a neural network model, consumers can gain a deeper understanding of the rental market, empowering them to make informed decisions regarding rent-related matters. This, in turn, enables landlords and tenants to make more intelligent choices, ultimately leading to an improvement in rental market efficiency. The increased

*Corresponding author: e-mail: daihongmei@shzq.edu.cn
**Corresponding author: e-mail: cfyang@nuk.edu.tw
https://doi.org/10.18494/SAM4594

ISSN 0914-4935
https://myukk.org/

accuracy provided by neural network predictions enables a more efficient allocation of resources within the rental market, further enhancing its overall efficiency. Moreover, the application of neural network technology in rent prediction can aid the real estate industry in effectively adapting to market fluctuations, consequently fostering its growth.

The synergy between neural networks and IoT is of profound importance, as neural networks are poised to be a driving force in elevating the capabilities and intelligence of IoT devices and systems.[3,4] Likewise, the relationship between AI and IoT is deeply intertwined, with AI serving as a linchpin in enhancing the potential of IoT systems.[5,6] This groundbreaking development marks a paradigm shift in IoT technology, introducing the transformative effect of deep learning into IoT device ecosystems. This pivotal advancement holds the promise of bringing the transformative power of AI to everyday household appliances, not only enriching their functionality but also fortifying data security and optimizing energy efficiency.

While neural networks can indeed contribute to improving rent market transparency, it is essential to consider potential drawbacks. One concern is the potential emergence of market monopolies facilitated by large real estate developers and financial institutions leveraging neural networks to predict rents solely for increasing their profits. In the absence of information asymmetry, tenants and landlords could face missed opportunities or unfair treatment. Another consideration is the potential invasion of personal privacy, as neural networks require extensive data for training purposes. The collection of such data may raise concerns about the unauthorized access or misuse of personal information. It is crucial to balance the benefits of using neural networks for rent prediction with the need for regulations and safeguards to address these concerns, ensure fair treatment and privacy protection, and minimize any negative impact on individuals or sectors affected by this technological shift.

The regression model developed by Chen *et al.* has demonstrated significant advancements in the field of rent prediction.[7] Building upon this progress, Islam *et al.* had introduced a deep learning approach tailored specifically for forecasting rental prices in online accommodation marketplaces like Airbnb.[8] Their method leveraged the power of deep learning techniques to enhance the accuracy of predictions in this domain. In a different vein, Oshodi *et al.* proposed an alternative approach of employing neural networks for rent price prediction.[2] They explored the potential of neural networks in capturing complex patterns and relationships within rental data, enabling more precise predictions. Recognizing the temporal dynamics of rental prices, Lazcano *et al.* proposed a recurrent neural network (RNN) approach that effectively incorporates the time component into rent forecasting.[9] This approach recognizes the importance of considering historical trends and patterns to improve the accuracy of predictions. Furthermore, Zulkifley *et al.* conducted a comparative study to evaluate the feasibility of using various machine learning algorithms, including neural networks, in house price prediction.[10] The results of their research offer insights into the performance and suitability of different algorithms in this specific domain, providing valuable guidance to researchers and practitioners.

From the aforementioned insights, it becomes evident that artificial intelligence and deep learning techniques have the potential to be applied in predicting a wide range of anticipated outcomes. Consequently, the central objective of this research is to harness the power of artificial intelligence and deep learning techniques to train a neural network specifically designed for the

prediction of house rents. By employing this machine learning pipeline, we can utilize Python along with machine learning libraries such as TensorFlow to train a neural network for rent prediction.[11] Key stages in this process encompass data collection, data preprocessing, dataset partitioning, model construction, model training, model optimization, model assessment, and model deployment. While proficiency in foundational machine learning concepts, such as appropriate selection of neural network models and tuning hyperparameters, is necessary, this pipeline offers an effective solution to the rent prediction challenge. Moreover, the application of training a neural network for house rent prediction demonstrated in this study has broader applicability across various domains and thus contributes to the development of robust machine learning models.

2. Analyses

Training a neural network to predict rent involves the following steps.

2.1 Data collection

We obtained housing rent data for the Zhongshan District, Taipei City, from the "591 Housing Trading Network", as Fig. 1 shows. The dataset comprises 689 entries with four key features: "Floorspace" (number of square feet), "Floor" (number of floors), "Use as a restaurant allowed", and "Pet allowed", and the corresponding "House-rent prices" are also provided. The dataset is sourced from the 591 Housing Transaction Network (in Chinese) and can be accessed at the following address: https://www.591.com.tw/. The collected data serves as the foundation for our research on rent prediction.

2.2 Data preprocessing

The collected data must be transformed into a suitable format for training the neural network. This involves several steps, including data cleaning, transformation, and scaling. For example, let us consider two candidates from among the housing search results, and we represent their

Fig. 1. (Color online) Housing search conditions of the 591 Housing Transaction Network.

data as follows: [9, 3, 1, 0] and [16999 NT$] for the first example and [6, 5, 0, 0] and [9000 NT$] for the second example. These data indicate that the first example is an indoor space with an area of 9 m^2 on the 3rd floor. It is permissible to operate a restaurant in this space but not to keep a pet, and the monthly rent for this property is 16999 NT$. Similarly, the second example is to a 6 m^2 space situated on the 5th floor. It is not possible to operate a restaurant or keep a pet, and the monthly rent for this property is 9000 NT$. By performing these preprocessing steps, the collected data is transformed into a format that can be effectively utilized by the neural network for training and prediction purposes.

2.3 Data segmentation (training set, validation set, and test set)

Before training a neural network, it is common practice to divide the data into three sets: training set, validation set, and test set. This division analogy can be compared with the study process of students, where the training set is akin to sample questions, the validation set is similar to exercises, and the test set represents final exams. During the training phase, the neural network only has access to the training set, just as students rely on sample questions to practice and learn. The validation set is used to assess the network's performance and simulate testing throughout the training process. It allows us to monitor the network's progress and identify any deviations from the desired training direction. If any significant deviations occur, the training can be halted immediately. The test set, on the other hand, is reserved for evaluating the final performance of the trained network. It remains unseen by the network during the training phase and is only used after the completion of all training cycles. This separation ensures that the network's performance is assessed on unseen data, similar to how students face a comprehensive examination. By employing both validation and test sets, we can verify the effectiveness of the trained neural network, ensuring that it can effectively solve problems and generalize well beyond the training data. This segmentation approach enables thorough evaluation and helps in optimizing the neural network's performance.

2.4 Data normalization

Each feature in the dataset may have different attributes and ranges. For instance, the "area" feature may have values ranging from 4 to 26, while the "Pet allowed" feature only takes the value 0 (no) or 1 (yes). The difference between 1 (yes) and 0 (no) already represents the maximum change for the "Pet allowed" feature, whereas for the "Area" feature, a change of 1 corresponds to a difference of 4 ft^2. However, for the "Number of pings" feature, a change of 1 only corresponds to the difference from 4 pings to 5 pings. Currently, larger values have a greater impact on the weight, and feeding these values directly into the neural network can lead to suboptimal training outcomes. Hence, it is essential to normalize the data before they are input into the neural network to ensure that each feature adheres to the same "measurement standard". In this study, a method called "standardization" was utilized for data normalization. It involves subtracting the mean from each data point and then dividing it by the standard deviation. By performing this operation, the data is centered around 0, with the standard deviation serving as the unit of measurement.

In the data normalization, the mean and standard deviations used for standardization should only be calculated for the training set, not the entire dataset. This approach ensures that the validation and test sets, which are used to evaluate the model's performance, remain independent. If the average and standard deviation are calculated using all the data, it could lead to data leakage. Data leakage occurs when information from unknown samples is inadvertently leaked into the model during training, resulting in inflated performance metrics during subsequent testing. In addition to normalizing the features, it is also important to normalize the labels. Normalizing the labels addresses the issue of varying scales and additionally facilitates the training process of neural networks. One approach to label normalization is to divide the labels by their maximum value, which allows for easier restoration in subsequent steps. This normalization ensures that the label values are scaled down, making them more suitable for effective neural network training.

2.5 Building a model–the sequence model

Keras, a deep learning development tool in Python, enables efficient and swift construction of neural networks (refer to Fig. 2). The neural network is constructed by a layer-by-layer approach, similar to assembling building blocks. Keras supports various types of content, including images, text, and sound. It allows the selection of different architectures such as CNN or RNN, without requiring any code modifications when switching hardware environments, such as CPUs and GPUs. This flexibility enables developers to leverage the computational power of different hardware configurations without significant modifications to the codebase. By utilizing the sequential model approach in Keras, developers can efficiently construct neural networks and explore different architectures for a wide range of applications in computer vision, natural language processing, audio processing, and more. When stacking neural layers in Keras, there is an alternative approach to using the **add()** method. Instead, each neural layer can be directly specified in sequence as parameters when constructing the model object. This method eliminates the need for explicit **add()** statements. The rectified linear unit (ReLU) is commonly

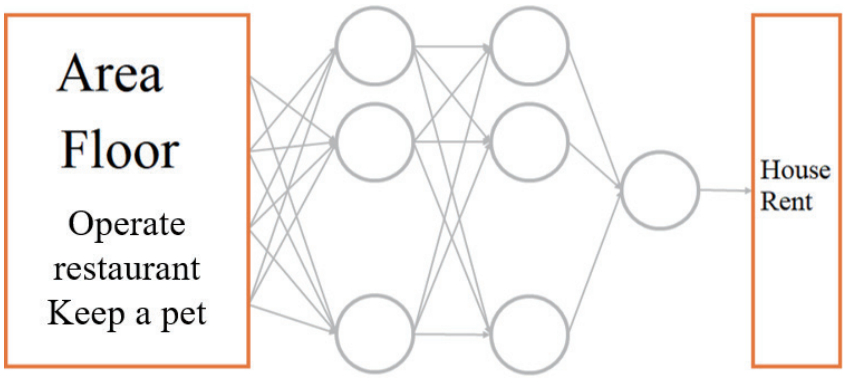

Fig. 2. (Color online) Establishing correspondence function between two groups of data-regression problem.

employed in hidden layers, and the specific definition of the ReLU activation function is defined as Eq. (1) and can be inferred from Fig. 3. Using this method, one can easily specify the architecture of the neural network by sequentially adding layers and specifying their properties, such as the number of units, input shape, and activation functions.

$$f(x) = \max(0, x) \tag{1}$$

2.6 Model compilation

Compiling the model involves specifying the training instructions for the model, which includes defining the loss function, evaluation metrics, and optimizer. Keras, as TensorFlow's high-level API, provides a convenient framework for constructing and training deep learning models. It supports rapid prototyping, advanced research, and deployment in production environments. In Keras, the loss function and metrics function are typically inversely related, meaning that a better evaluation result corresponds to a smaller loss value. Therefore, Keras allows us to use the loss function directly as the opposite interpretation of the metrics function. The metrics returned by the evaluation are solely used for performance evaluation and are not utilized during the training process. The optimizer relies solely on the loss value returned by the loss function for optimization, which is unrelated to the evaluation of performance. The optimizer plays a crucial role in accelerating the training of neural networks and improving the overall training effectiveness. It accomplishes this by transforming the gradient and subsequently applying the recalculated update values to the weight and bias parameters. Adaptive moment estimation (Adam) optimization is a stochastic gradient descent method that combines the benefits of momentum and adaptive gradient (AdaGrad) optimizers. It achieves this by adaptively estimating first-order and second-order moments. Adam maintains an exponentially decaying average of past gradients, similarly to momentum, and also keeps a squared decaying average of past gradients. The combination is achieved as follows.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)(\partial L_t / \partial W_t) \tag{2}$$

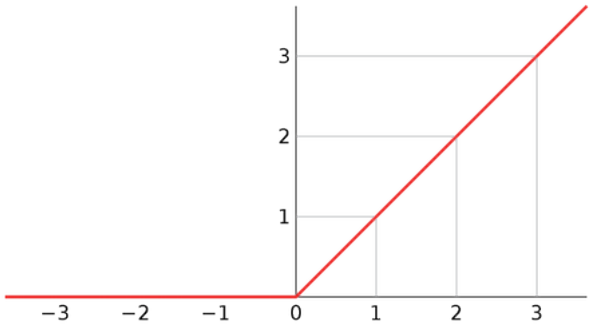

Fig. 3. (Color online) Activation function of ReLU.

$$v_t = \beta_2 m_{t-1} + (1-\beta_2)(\partial L_t / \partial W_t)^2 \tag{3}$$

Additionally, Adam performs "deviation correction" for the parameters, which ensures that each learning rate falls within a specific range. This facilitates the updating of parameters and the bias correction for m_t and v_t. Consequently, the parameter updates become more stable and reliable.

$$\hat{m}_t = m_t / \left(1-\beta_1^t\right) \tag{4}$$

$$\hat{v}_t = v_t / \left(1-\beta_2^t\right) \tag{5}$$

The Adam weight update equation as follows.

$$W \leftarrow W - \eta\left[\hat{m}_t / \left(\sqrt{\hat{v}_t} + \epsilon\right)\right] \tag{6}$$

2.7 Training the model

Once the model is compiled, the training process can be initiated by executing the **fit()** function. During training, several common parameters come into play. At the conclusion of each training cycle, the validation data are utilized to calculate the validation loss and assess the model's performance on unseen data. It is important to note that this information is not employed for training purposes, but rather, serves as an evaluation metric for assessing the model's performance.

2.8 Outcome evaluation–the loss curve

The objective of a neural network is to minimize the loss value. Plotting the loss curve provides a clear visualization of whether the neural network is progressing towards this goal. In the case of regression losses, the mean absolute error (MAE) is commonly used. MAE is defined as the average of the absolute differences between the actual and predicted values. By analyzing the loss curve, one can assess the performance and convergence of the neural network.

3. Results and Discussion

The objective of a neural network is to minimize the loss value of the training data. Figure 4 depicts the loss curves for both the training data and the validation data. It is evident from the figure that the loss value of the validation data is lower than that of the training data, indicating that the neural network model is effectively progressing towards its goal. By reducing the loss

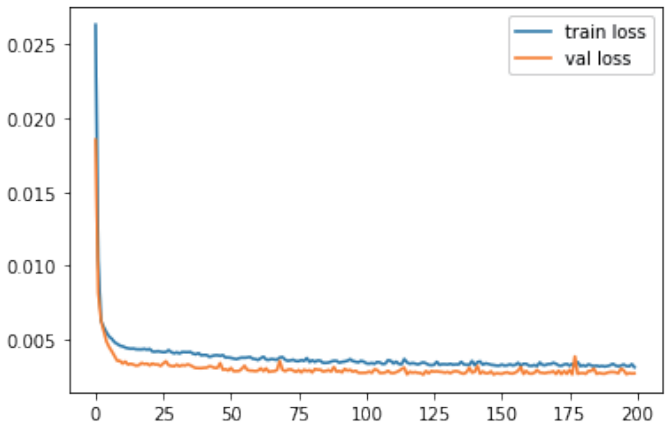

Fig. 4. (Color online) Training loss and validation loss.

value, the neural network improves its ability to accurately predict the outcomes. This is crucial for achieving optimal performance and enhancing the model's generalization capabilities. The decreasing trend of the loss curve signifies that the model is learning and adapting to the training data, demonstrating promising progress in its training process.

Once the housing rent prediction model is obtained, it can be utilized to make rent predictions, as depicted in Fig. 5. The blue bars represent the actual rent values, while the orange bars represent the predicted rent values. The *x*-axis corresponds to the number of data points, and the *y*-axis represents the rent in Taiwanese dollars. To assess the accuracy of the predictions, 30 test data points are input into the model. By comparing the predicted rent values with the corresponding actual rent values, the difference between the two is calculated, resulting in the generation of an error distribution map, as shown in Fig. 6. The error distribution map provides valuable insight into the disparities between the predicted and actual rent values, enabling a comprehensive analysis of the model's performance. Visualizing the magnitude and direction of the prediction errors facilitates a deeper understanding of the model's predictive capabilities.

However, relying solely on Fig. 5 to assess the quality of the forecast can be challenging. To overcome this limitation, the error distribution map in Fig. 6 is employed to conduct a more detailed analysis of the forecast accuracy. By examining the distribution of errors across different ranges on the *x*-axis (error range in Taiwanese dollars) and the corresponding number of data points on the *y*-axis (number of houses), a clearer understanding of the model's performance and the prevalence of prediction errors can be obtained. This comprehensive evaluation helps to refine the model and enhance its predictive capabilities.

To analyze the accuracy of the predictions, the "error value" is calculated by subtracting the predicted rent from the actual rent. This error value range is then divided into 15 equal parts, and the number of data points falling within each error range is determined. The *x*-axis of the error distribution map represents the error range in Taiwan dollars, while the *y*-axis represents the number of data points (number of houses). By examining the error distribution map, it becomes evident that the majority of the error values fall within the range of ±NT$2000. Only a few data points exhibit relatively large errors. This visualization provides a clear understanding of the distribution and magnitude of errors in the rent predictions.

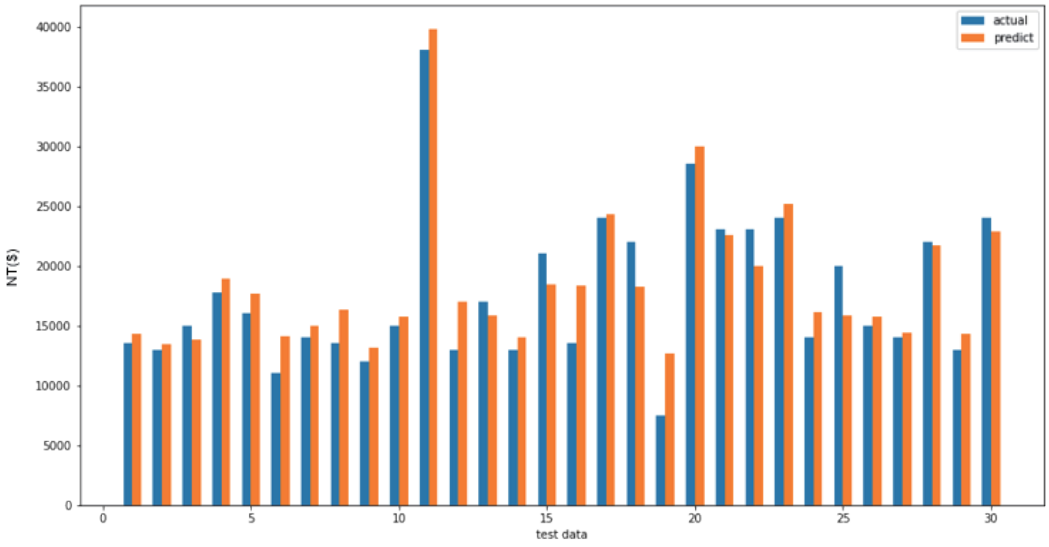

Fig. 5. (Color online) Comparison chart.

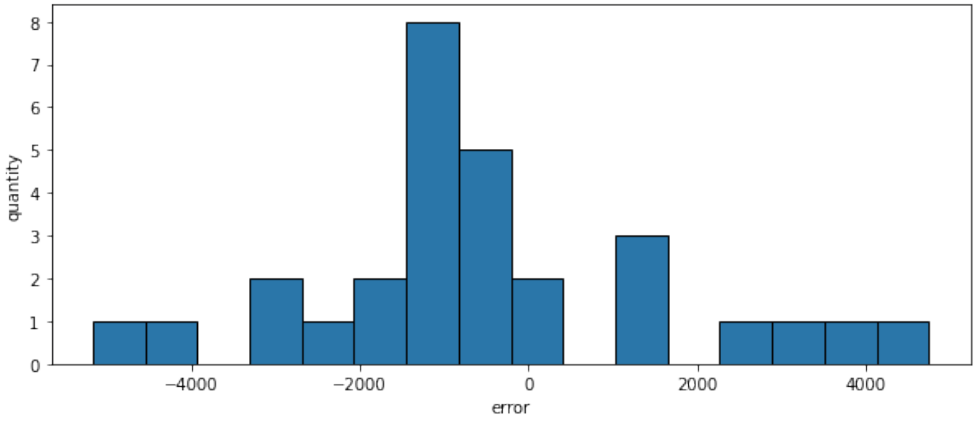

Fig. 6. (Color online) Error distribution map.

4. Conclusions

In this study, we utilized the 591 Housing Trading Network as our data source, extracting five key variables: the area in ft^2, floor level, permissibility of operating a restaurant, permissibility of keeping a pet, and the corresponding rent. Our objective was to construct a neural network model capable of accurately predicting the rental price based on the first four input features. Through our experiment, we found that by employing a regression prediction model and utilizing only these four features, we achieved remarkably accurate predictions of the rental prices. This outcome suggests that these specific variables hold significant predictive power in determining the rent associated with a given property. By leveraging the power of neural networks and focusing on these crucial features, our model can effectively estimate rental prices reliably and accurately. The results of this research demonstrate the potential of using a reduced set of inputs to achieve accurate predictions in the realm of rent forecasting.

Acknowledgments

This work was supported by the Foreign Cooperation Project of Fujian Provincial Department of Science and Technology (Grant No. 2020I0022), MOST 109-2221-E-390-023, MOST 110-2622-E-390-002, and MOST 110-2221-E-390-020.

References

1 X. Xu and Y. Zhang: Intell. Syst. Appl. **12** (2021) 200052.
2 O. Oshodi, I. Ohiomah, T. Odubiyi, C. Aigbavboa, and W. Thwala: Proc. 11th Inter. Conf. Construction in the 21st Century, London (2019) 309–313.
3 M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque: Simul. Model. Pract. Theory **101** (2020) 102031.
4 Z. Ahmad, A. S. Khan, K. Nisar, I. Haider, R. Hassan, M. R. Haque, S. Tarmizi, and J. J. P. C. Rodrigues: Appl. Sci. **11** (2021) 7050.
5 A. Ghosh, D. Chakraborty, and A. Law: CAAI Trans. Intell. Technol. **3** (2018) 208.
6 G. Rathee, A. Khelifi, and R. Iqbal: Wirel. Commun. Mob. Comput. **2021** (2021) 5754322.
7 D. Chen, F. Hu, G. Nian, and T. Yang: Entropy **22** (2020) 193.
8 Md D. Islam, B. Li, K. S. Islam, R. Ahasan, Md R. Mia, and Md E. Haque: Mach. Learn. Appl. **7** (2022) 100208.
9 A. Lazcano, P. J. Herrera, and M. Monge: Mathematics **11** (2023) 224.
10 N. H. Zulkifley, S. A. Rahman, N. H. Ubaidullah, and I. Ibrahim: Inter. J. Mod. Educ. Comput. Sci. **6** (2020) 46.
11 L. Schwinn, R. Raab, A. Nguyen, D. Zanca, and B. Eskofier: **53** (2023) 19843.