Research Article

Nimet Isik* and Omer Can Eskicioglu

Predictability of machine learning framework in cross-section data

https://doi.org/10.1515/phys-2022-0261
received April 01, 2023; accepted June 10, 2023

Abstract: Today, the use of artificial intelligence in electron optics, as in many other fields, has begun to increase. In this scope, we present a machine learning framework to predict experimental cross-section data. Our framework includes 8 deep learning models and 13 different machine learning algorithms that learn the fundamental structure of the data. This article aims to develop a machine learning framework to accurately predict double-differential cross-section values. This approach combines multiple models such as convolutional neural networks, machine learning algorithms, and autoencoders to create a more robust prediction system. The data for training the models are obtained from experimental data for different atomic and molecular targets. We developed a methodology for learning tasks, mainly using rigorous prediction error limits. Prediction results show that the machine learning framework can predict the scattering angle and energy of scattering electrons with high accuracy, with an R-squared score of up to 99% and a mean squared error of <0.7. This performance result demonstrates that the proposed machine learning framework can be used to predict electron scattering events, which could be useful for applications such as medical physics.

Keywords: differential cross section, machine learning algorithm, regression algorithms, autoencoders, deep learning algorithm

1 Introduction

Electron–atom/molecule collisions have been extensively studied over the past century and remain an important area of research due to their wide range of applications in various fields [1–7]. The study of electron impact ionization has contributed significantly to our understanding of the fundamental properties of atoms and molecules and ionization dynamics [8–13]. In mass spectrometry, electron ionization is employed to generate ions from gaseous samples, which are then separated and detected based on their mass-to-charge ratio [14,15]. Electron impact ionization experiments allow for the study of reaction mechanisms and kinetics of organic molecules [16–20]. In plasma physics, electron ionization plays a crucial role in initiating and maintaining plasmas, which are ionized gases with unique properties [21,22]. In addition, electron ionization has implications in atmospheric physics, astrophysics, and radiation physics [23–25]. The ongoing research in electron ionization focuses on advancing our knowledge of the underlying mechanisms, exploring new applications, and developing more accurate theoretical models and computational methods to describe ionization processes.

In ionization dynamics, cross-section measurements provide valuable insights into the ionization dynamics and help understand the underlying processes involved. Single differential cross-section (SDCS), double differential cross-section (DDCS), and triple differential cross-section (TDCS) measurements provide information about the energy and angular distributions of the emitted electrons, offering insights into the dynamics of ionization reactions. The SDCS represents ionization probability as a function of the scattering angle or the energy of the ejected electron while integrating over other variables. It provides information about the angular distribution and energy dependence of the ionization process [3]. The DDCS describes ionization probability as a function of both the energy and scattering angle of the ejected electron. It provides more detailed information about the energy-angular correlation of the ionized electrons [2,26]. The TDCS represents ionization probability as a function of the energy and scattering angles of both the ejected and scattered electron. TDCS measurements provide a more comprehensive picture of the ionization process by considering the correlations between the outgoing and scattered electrons [8–13]. These measurements allow researchers

* **Corresponding author: Nimet Isik,** Mathematics, and Science Education Department, Burdur Mehmet Akif Ersoy University, Burdur, Turkey, e-mail: nokumus@mehmetakif.edu.tr
Omer Can Eskicioglu: Software Engineering Department, Burdur Mehmet Akif Ersoy University, Burdur, Turkey

to explore the complete scattering dynamics and energy transfer involved in the ionization reaction.

The accuracy and reliability of the models can be evaluated by comparing the theoretical calculations with the experimental data. Inconsistencies between experimental measurements and theoretical calculations may highlight areas where the theoretical model needs improvement, such as incorporating additional physical effects or refining computational methodologies [27,28].

In the last decade, the burgeoning development of artificial intelligence (AI) has begun transforming many aspects of science. AI has been introduced as an alternative solution to many problems encountered in optics [29–34], atomic and molecular physics [35–42], and high-energy physics [43–46]. AI-based techniques have been used to predict a complex relationship between input and output data. It is one of the most promising techniques used to improve the understanding and utilization of data. AI-based algorithms are quite capable of predicting and classifying physics data, but their limitations must be considered [47]. These algorithms typically require large amounts of training data to generalize and make accurate predictions. Limited data may lead to overfitting or under-generalization of the model [48].

Machine learning (ML) algorithms are a subset of AI. These algorithms can learn by drawing meaningful conclusions from specific data [49]. ML algorithms are used to recognize patterns in large datasets and make decisions based on those patterns. ML algorithms are trained using large amounts of data and powerful computing resources. ML algorithms aim to create models that can learn complex tasks from data without relying on task-specific rules. By using these algorithms, computers learn from past experiences and improve performance without being explicitly programmed [47,48]. In recent years, ML algorithms have shown great potential in addressing complex prediction problems in optics [34] and references therein and high energy physics [43–45]. ML algorithms have emerged as promising tools for accurate predictions of cross-section values in atomic and molecular physics [38,41]. By leveraging large datasets and powerful computational algorithms, ML models can capture complex patterns and relationships in data, enabling more accurate predictions.

In atomic and molecular physics, experimental observations are often compared to theoretical models to validate our understanding of the underlying physical processes. Theoretical predictions rely heavily on the accurate prediction of cross-section values, as they provide crucial information about the probabilities of different interactions occurring. By comparing experimental data to theoretical predictions, physicists can test the validity of existing theories or search for new physics phenomena. The accurate prediction of cross-section values for particle interactions is a crucial aspect of physics research. However, predicting cross-section values accurately is a highly complex task. It involves considering numerous factors, such as the properties of the interacting particles, their energies, and the specific conditions under which the collision occurs. Traditional analytical methods often struggle to capture the intricate relationships between these variables, leading to limited prediction accuracy. Leveraging this potential, the article aims to develop a comprehensive ML framework capable of accurately predicting cross-section values for different target atom/molecule interactions with electrons.

To achieve this, we propose integrating multiple ML models, convolutional neural networks (CNNs), and autoencoders into a unified framework. CNNs are well-suited for analyzing and extracting features from complex, high-dimensional datasets, making them an ideal choice for processing particle interaction data. In contrast, autoencoders provide powerful unsupervised learning capabilities, enabling the discovery of latent representations that capture essential characteristics of the input data [46]. By combining the strengths of CNNs and autoencoders, we aim to create a more robust prediction system that can effectively capture the intricate relationships between input variables and the corresponding cross-section values. This integration will enhance the model's ability to generalize and make accurate predictions.

In each regression algorithm, hyper-parameter optimization is performed using the grid-search method. The data used for training the models were obtained from experimental data of particle interactions in various atomic and molecular targets, including helium (He), neon (Ne), argon (Ar), krypton (Kr), xenon (Xe), nitrogen (N_2), oxygen (O_2), hydrogen (H_2), methane (CH_4), water (H_2O), carbon monoxide (CO), ammonia (NH_3), and carbon dioxide (CO_2) [2]. There are several important steps to consider to ensure the generalizability of ML algorithms when preparing training datasets. The first of these is data collection. It should cover all possible inputs and outputs the algorithm may encounter for the problem under consideration. The dataset should be cleaned by removing irrelevant or duplicate data points, correcting errors, processing missing values, and addressing outliers. Data cleaning ensures that the algorithm is trained on high-quality and reliable data. Removing irrelevant or redundant features can increase the efficiency and generalizability of the model. Normalizing the input data to a common scale or distribution ensures that the algorithm is not biased towards features with larger magnitudes and helps improve convergence during training. In addition,

K-fold cross-validation to evaluate the algorithm's performance on multiple subsets of data help evaluate generalization ability and reduce over-fitting by averaging performance measures across different data segments. We will also demonstrate its effectiveness by conducting extensive experiments and evaluating its performance against existing prediction methods. By combining ML, CNNs, and autoencoder, we strive to provide physicists with a valuable tool to enhance their understanding of particle interactions and support future discoveries in the field.

1.1 ML framework

ML uses a layered architecture of algorithms to progressively extract higher-level features from raw data. It is made up of multiple layers of computational units that are organized hierarchically. At the lowest level, the input data is processed, and then the output is used to modify the parameters of the next layer, and so on, until the desired outcome is obtained [49,50].

The data used for the ML algorithm were a set of experimental differential cross-section (DCS) data. The experimental data set collected by Opal and co-workers was used as the basis of this work [2]. The experimental data point is characterized by a set of descriptors for the normalized results of relative measurements of electron-production cross-section differentials in angle over the 30 and 150° range and in ejected energy over the 4–200 eV range. Primary electron energies are 500 eV for all data. The inputs for the ML are the coordinates and energies of the scattered electrons.

The architecture of an ML model can vary greatly depending on the purpose and type of problem that it is trying to solve. Generally, the model consists of an input layer, multiple hidden layers, and an output layer. Each layer is made up of neurons that are connected, and the connections between the neurons are weighted. The weights are adjusted over time as the model is trained to refine its predictions. Once the architecture is defined, the model can be compiled using an appropriate optimizer and loss function. An optimizer is an algorithm used to adjust the weights of the model to improve its performance. Commonly used optimizers include stochastic gradient descent, Adam, and RMS prop. The loss function is used to measure the accuracy of the model's predictions and is usually chosen based on the type of problem being solved. We used loss functions including mean square error (MSE), root mean square error (RMSE), mean absolute error (MAE), and R-squared (R^2) score. MSE is a measure of the average of the squares of the errors or deviations from the actual value. It measures how close a predicted value is to the actual value. RMSE measures how far off predictions are from the actual values. It is calculated by taking the square root of the MSE. MAE measures the average magnitude of errors in a set of predictions without considering their direction. It is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. In ML, the R^2 score, also known as the coefficient of determination, is a statistical measure that assesses the goodness-of-fit of a regression model to the observed data. It indicates the proportion of the variance in the dependent variable that the independent variables in the model can explain. The R^2 score is calculated using the formula $R^2 = 1 - (\mathrm{SSR}/\mathrm{SST})$, where SSR (sum of squares residual) represents the sum of the squared differences between the predicted values and the actual values, and SST (total sum of squares) represents the sum of the squared differences between the actual values and the mean of the dependent variable. The R^2 score can range from 0 to 1. A score of 1 indicates a perfect fit, where the regression model precisely predicts the dependent variable. A score of 0 means that the model does not provide any improvement over simply predicting the mean of the dependent variable. R^2 score is a measure that indicates how close data points are to a fitted regression line. The higher this score, the better our model fits our data.

A validation dataset is an important tool for evaluating the performance of a machine-learning model. It measures how well a model can predict unseen data which is not part of the training dataset. The validation dataset is a subset of the training dataset held back from the model during the training process. The model is then tested on the validation dataset to evaluate its performance. The performance of the model is measured using various metrics such as RMSE, MSE, MAE, and R^2 score.

Tuning hyperparameters is an important and necessary step in optimizing the performance of the model. Hyperparameter tuning is the process of optimizing the values of the hyperparameters of a model to improve its performance. This can involve testing different combinations of hyperparameter values to determine which combination yields the best results. It is important to note that hyperparameter tuning should be done systematically, as random changes to hyperparameters can lead to suboptimal performance and wasted time. By tuning the hyperparameters of a model, we can ensure that the model is performing at its best and will generate the best results for our data.

The model is tested with the data in the test set. The test aims to evaluate how well the model generalizes to

unseen data. During the test, the model should be evaluated on both the training and testing datasets. The results should then be compared to determine if the model has improved and is ready for deployment. At the end of the test, the algorithm with a low error value is selected by considering the error metrics.

Finally, the model was deployed on real-world data, and the results were compared with the expected results. In Table 1, we present the introduction steps of the ML framework:

In our proposed study, autoencoder is used to increase the performance of the ML framework. Autoencoders have different usage areas. Our study provides the extraction of important features from the data set. Autoencoders consist of three layers: input layer, hidden layer, and output layer. The purpose of the architecture used is to reveal important attributes directly or indirectly related to the result. It was used in the data preparation stage before giving the dataset to the ML algorithms. Autoencoders are neural networks with equal input and output layers. It takes the neural network input data and makes bottlenecks in the hidden layer to extract important features. In autoencoders, the section between the input layer and the hidden layer is called the Encoder, and the section between the hidden layer and the output layer is called the Decoder. It tries to obtain a more understandable and precise result by reconstructing the input data. Figure 1 shows the structure of the autoencoder that is used in our study. In the pre-training model summary of the created deep learning model, the structure of the model, output shapes of layers, the total number of parameters, and the values of learnable parameter numbers are obtained. In Figure 1, information is obtained about the layers of this structure and the output shape of these layers. Since the input and output values in the layers shown in Figure 1 are a dimension of the data to be imported, the pre-training model output is written as “None.”

In our study, training was also conducted using different deep learning models such as Mobilenetv1, Mobilenetv2, Mobilenetv3-Small, MobilenetV3-Large, VGG11, VGG13, VGG16, and VGG19 deep learning architectures. Figure 2 shows the flow chart of the study. In our study, by rearranging the dimensions of the models, our dataset was brought to a one-dimensional state where it would be appropriate. The Adam optimization algorithm was used in the models. In each model, the most efficient hyperparameters were used with the grid-search method.

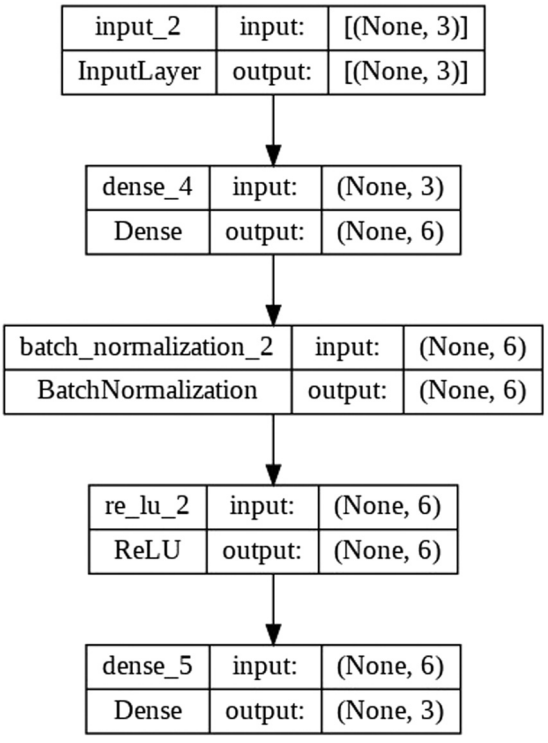

Figure 1: Proposed autoencoder architecture in our study.

2 Results

2.1 Regression algorithm results

ML frameworks were trained using 8,415 data in our dataset. Each data point in a dataset typically corresponds to a specific combination of secondary energy, angle, and

Table 1: Proposed ML framework steps

1)	Gather data: Collect relevant data that can be used to train the ML and deep learning model
2)	Pre-process data: Clean, normalize, and prepare the data for the model
3)	Select an algorithm: Select the best algorithm for the problem
4)	Train the model: Feed the pre-processed data into the model to allow it to learn
5)	Evaluate model performance: Test the model’s accuracy on unseen data to measure its performance
6)	Optimize model: Tune the model parameters to improve performance

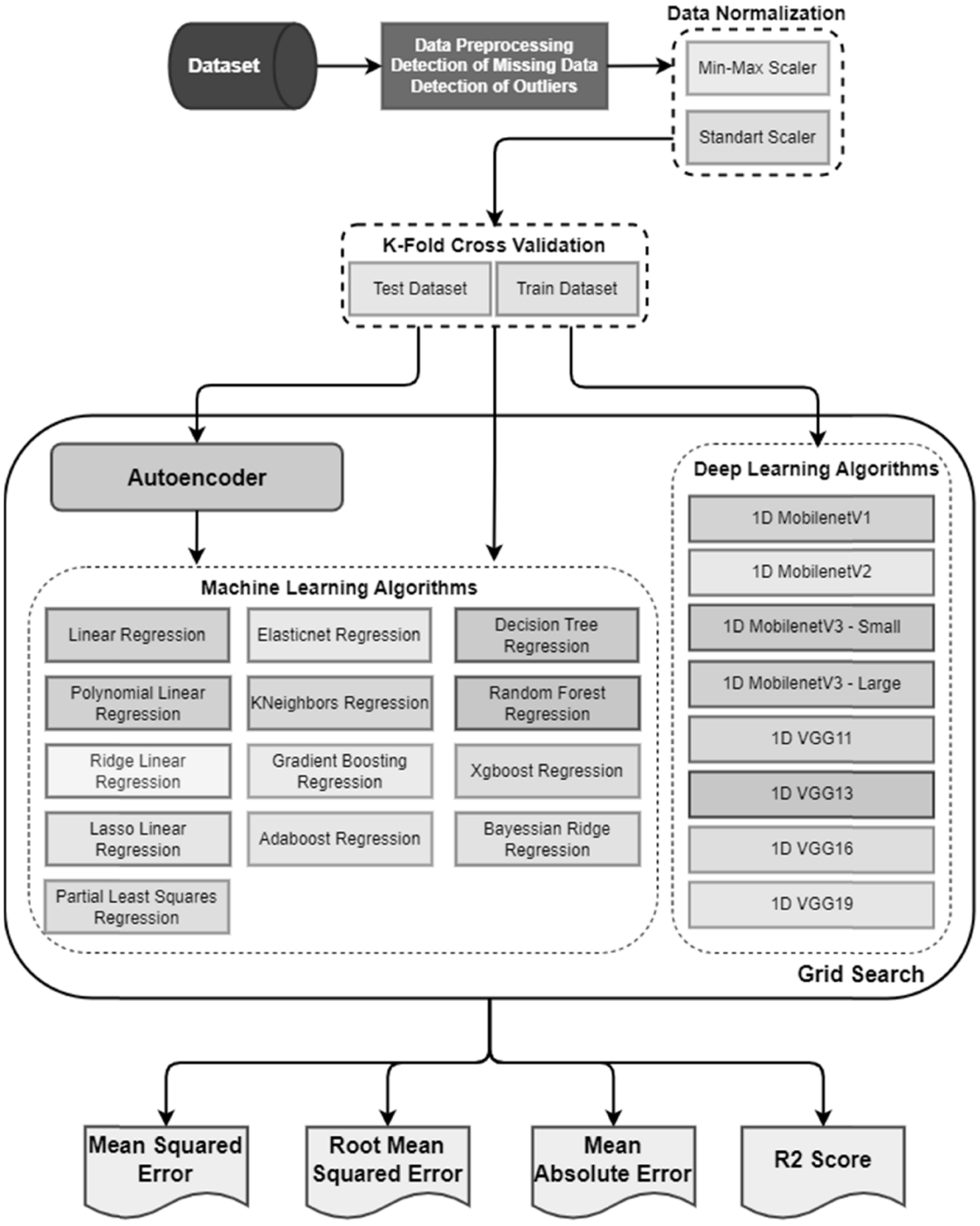

Figure 2: Flow chart of our proposed ML framework for predicting cross-section data.

associated DCS values. Data for a particular gas target were recorded in a systematic way for large-scale energy and angular distributions of electrons ejected in the collisions between electrons and some atoms and simple molecules (He, Ne, Ar, Kr, Xe, N_2, O_2, H_2, CH_4, H_2O, CO, NH_3, and CO_2). We evaluated our model on a separate test set consisting of 2,000 data from different targets to avoid the risk of overfitting. In this way, we construct an accurate and precise DCS predictor. The purpose of the test dataset is to assess the model's performance on unseen data. This evaluation helps determine whether the model has successfully learned the underlying patterns and can accurately predict unseen data. When evaluating our model on the test set, the ML framework was evaluated using performance

Table 2: Regression algorithm performance results

Regression algorithm	MSE	RMSE	MAE	R^2 score
Linear regression	108.97	10.43	7.60	0.41
Polynomial linear regression	13.71	3.70	2.31	0.92
Ridge linear regression	108.92	10.43	7.59	0.41
Lasso linear regression	109.02	10.44	7.60	0.41
Elastic net regression	109.00	10.44	7.60	0.41
K nearest neighbors regression	29.90	5.46	3.81	0.83
Gradient boosting regression	**0.69**	**0.83**	**0.49**	**0.99**
Decision tree regression	93.24	9.65	6.30	0.49
Random forest regression	1.11	1.05	0.67	0.99
Xgboost regression	42.08	6.48	4.52	0.77
Adaboost regression	49.77	7.05	5.55	0.73
Partial least squares regression	108.97	10.43	7.60	0.41
Bayesian ridge regression	109.03	10.44	7.60	0.41

Gradient boosting regression (GBR) is the best regression model for predicting DCS data.

metrics. To assist interpretation of the results, the MSE, RMSE, MAE, and R^2 score values for each algorithm are obtained and presented in Table 2.

The result of these algorithms varies depending on the dataset used and the hyperparameters chosen. The quality, diversity, and representativeness of the training data play a crucial role in determining how well the algorithm will perform on unseen data. When training an ML model, the aim is to expose it to a diverse range of examples that capture the patterns and variations present in the target domain. By learning from a broad and representative dataset, the model can extract meaningful features and make accurate predictions on unseen data that shares similar characteristics with the training set. However, it is important to acknowledge that no dataset can perfectly represent all possible variations and scenarios that may be encountered in the real world. There is always a degree of uncertainty and potential for the model to encounter new patterns or situations that it has not encountered during training. To mitigate this, ML models are evaluated on their performance on the training data and their ability to generalize to unseen data. This is typically done by evaluating the model on a separate validation or test set that was not used during the training process. Models trained on specific datasets may perform well on similar data, but their performance may degrade when applied to significantly different or out-of-distribution data. Generally, the best results are obtained when the hyperparameters are tuned to the dataset and when a combination of algorithms is used. The results can also include the RMSE and the MAE of the model. We evaluated our model on a separate test set, achieving an R^2 score of 99%.

The results show that gradient boosting regression (GBR) is the best regression model for predicting experimental data. It's MSE is 0.69, RMSE is 0.83, MAE is 0.49, and R^2 score is 0.99. GBR is a powerful ML algorithm that combines multiple weak learners to create a strong learner. It is an ensemble method that uses decision trees as its base learners and sequentially combines them to create a more accurate prediction. GBR is more accurate than other regression models such as linear regression, support vector machines, and random forests. Additionally, GBR is more robust and less prone to overfitting than other models. Therefore, GBR is the best choice for predicting experimental data.

Figure 3a shows a comparison of the predicted values with the experimental test data. It shows that the predicted values closely match the experimental data, indicating that the prediction model is accurate and reliable. The results show that ML algorithms can accurately predict cross-

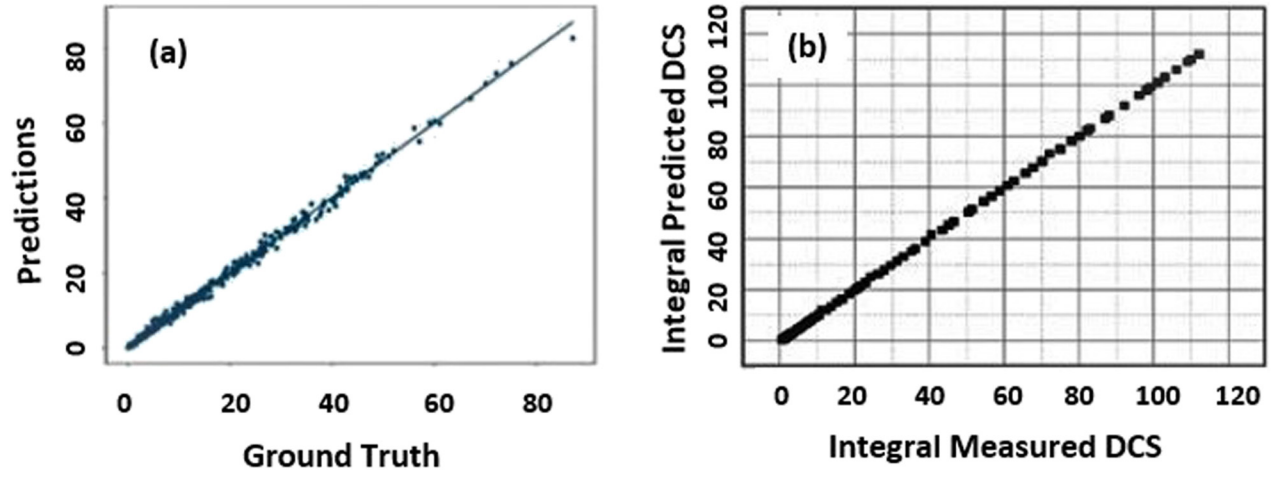

Figure 3: (a) Ground truth *vs* prediction scatter plot outlier. (b) The comparison of the integral of the predicted data with measured data for the He atom.

Table 3: Autoencoder + regression algorithm performance results

Autoencoder (A) + regression algorithm	MSE	RMSE	MAE	R^2 score
A + Linear regression	78.58	8.86	6.39	0.61
A + Polynomial linear regression	71.72	8.47	5.14	0.64
A + Ridge linear regression	78.57	8.86	6.39	0.61
A + Lasso linear regression	129.88	11.40	8.65	0.35
A + Elastic net regression	121.33	11.01	8.35	0.39
A + K nearest neighbors regression	**2.11**	**1.45**	**0.98**	**0.99**
A + Gradient boosting regression	64.37	8.02	3.85	0.68
A + Decision tree regression	127.33	11.28	8.44	0.36
A + Random forest regression	57.23	7.57	4.21	0.71
A + Xgboost regression	78.86	8.88	6.38	0.60
A + Adaboost regression	74.06	8.61	6.58	0.63
A + Partial least squares regression	97.18	9.86	7.22	0.51
A + Bayesian ridge regression	78.56	8.86	6.38	0.61

The autoencoder + KNN algorithm is the best model for predicting DCS data.

Table 4: Deep learning models' performance results

Deep learning models	MSE	RMSE	MAE	R^2 score
1D MobilenetV1	1375.57	37.08	20.12	0.73
1D MobilenetV2	**217.90**	**14.76**	**9.49**	**0.72**
1D MobilenetV3 – Small	299.91	17.31	8.61	0.62
1D MobilenetV3 – Large	376.83	19.41	9.83	0.52
1D VGG11	273.04	16.52	6.47	0.65
1D VGG13	226.28	15.04	6.41	0.71
1D VGG16	**177.47**	**13.32**	**7.01**	**0.77**
1D VGG19	306.49	17.50	12.49	0.61

VGG16 and MobilenetV2 models show acceptable results.

sections for a variety of atomic and molecular targets with high accuracy and precision. In Figure 3a, we present the performance evaluation results of an ML algorithm using K-fold cross-validation on a test dataset consisting of 2,000 samples. The performance metric, in this case, is reported as a result of the evaluation, with an error value of 0.09. K-fold cross-validation is a technique used to assess the performance and generalization capability of an ML algorithm. It involves dividing the dataset into K equally-sized subsets or "folds." The algorithm is trained on K-1 folds and evaluated on the remaining fold, repeating this process K times to ensure that each fold acts as the evaluation set once. This technique provides a robust estimate of the model's performance by averaging the evaluation results across all K iterations. The performance metric value of 0.09 shows the algorithm's predictive capability on the given test dataset. The specific interpretation and implications of this value may depend on the nature of the problem being addressed and the context of the study. Figure 3a

Table 5: GBR prediction and ground truth values for He DCS data

Ground truth	Prediction	Ground truth	Prediction	Ground truth	Prediction	Ground truth	Prediction
22.1	22.45	16.0	16.65	41.7	41.48	1.81	2.06
4.22	3.91	2.57	2.90	29.9	29.58	34.7	31.97
33.4	32.15	2.3	2.63	0.77	0.82	10.8	10.33
3.08	2.90	21.1	20.79	1.25	1.20	26.9	27.70
23.3	22.60	1.57	1.74	21.6	21.91	8.3	8.04
0.74	0.76	3.44	3.35	1.07	1.07	60.0	58.75
9.9	9.17	22.4	21.76	10.3	10.51	41.3	40.87
7.0	6.52	3.71	4.04	12.3	12.40	16.6	17.46
1.47	1.41	34.4	34.26	3.36	3.16	26.7	26.56
0.385	0.37	5.2	5.16	9.0	9.47	2.35	2.38
3.89	3.90	1.23	1.24	3.04	3.27	3.67	3.88
9.6	10.42	2.23	2.59	6.2	6.46	25.7	25.80
4.98	4.53	11.5	12.45	0.52	0.58	14.8	13.27
6.8	6.75	8.3	8.06	2.24	2.53	24.9	24.19
5.4	5.49	13.1	13.52	3.55	3.84	23.3	24.81
13.4	12.89	14.6	15.14	6.6	6.87	3.86	3.90
2.84	2.67	14.5	14.57	3.13	3.13	1.03	1.12
20.3	20.75	13.9	13.40	25.9	26.38	4.08	4.23
26.8	29.25	4.83	6.89	2.27	2.32	2.51	2.37
30.9	30.40	1.7	1.87	3.13	3.55	8.0	8.41
1.37	1.47	9.2	9.43	5.4	5.19	0.295	0.14
32.3	33.93	39.3	36.57	6.9	6.66	5.2	5.09
2.05	2.15	16.3	13.69	1.83	2.07	1.15	1.24
15.5	15.56	4.29	4.31	0.407	0.29	6.3	6.28
1.07	1.10	3.75	3.74	21.4	21.26	2.36	2.30

shows the effectiveness of the ML algorithm on the test dataset, as determined through K-fold cross-validation. It provides valuable insights into the algorithm's predictive performance. Figure 3b compares the integral of the predicted data and measured DCS values for the He target. By comparing the integral of the predicted and measured values, we can assess the accuracy and validity of our framework. Figure 3b indicates good agreement between the integral of the predicted and measured DCS values.

2.2 Autoencoder + regression algorithm performance results

The autoencoder is used to extract important data features and reduce the error rates of the regression algorithms. In the training process, the autoencoder learns the basic properties of the data and reflects these properties to the hidden layer. Thus, the autoencoder can highlight important features in the data and remove unnecessary details. In Table 3, autoencoder layers were added before the regression algorithms, and the architecture was updated. In this case, the autoencoder + KNN algorithm showed nearly the same result as the GBR. The results in Table 3 were obtained by using the autoencoder and regression algorithms sequentially. Although this structure increases the performance of some models to a certain extent, performance decreases have been observed in a few models. In general, increases were observed in the general average of the error metrics of all algorithms. It has been observed that the performance of the K-nearest neighbors (KNN) regression algorithm has increased significantly, increasing

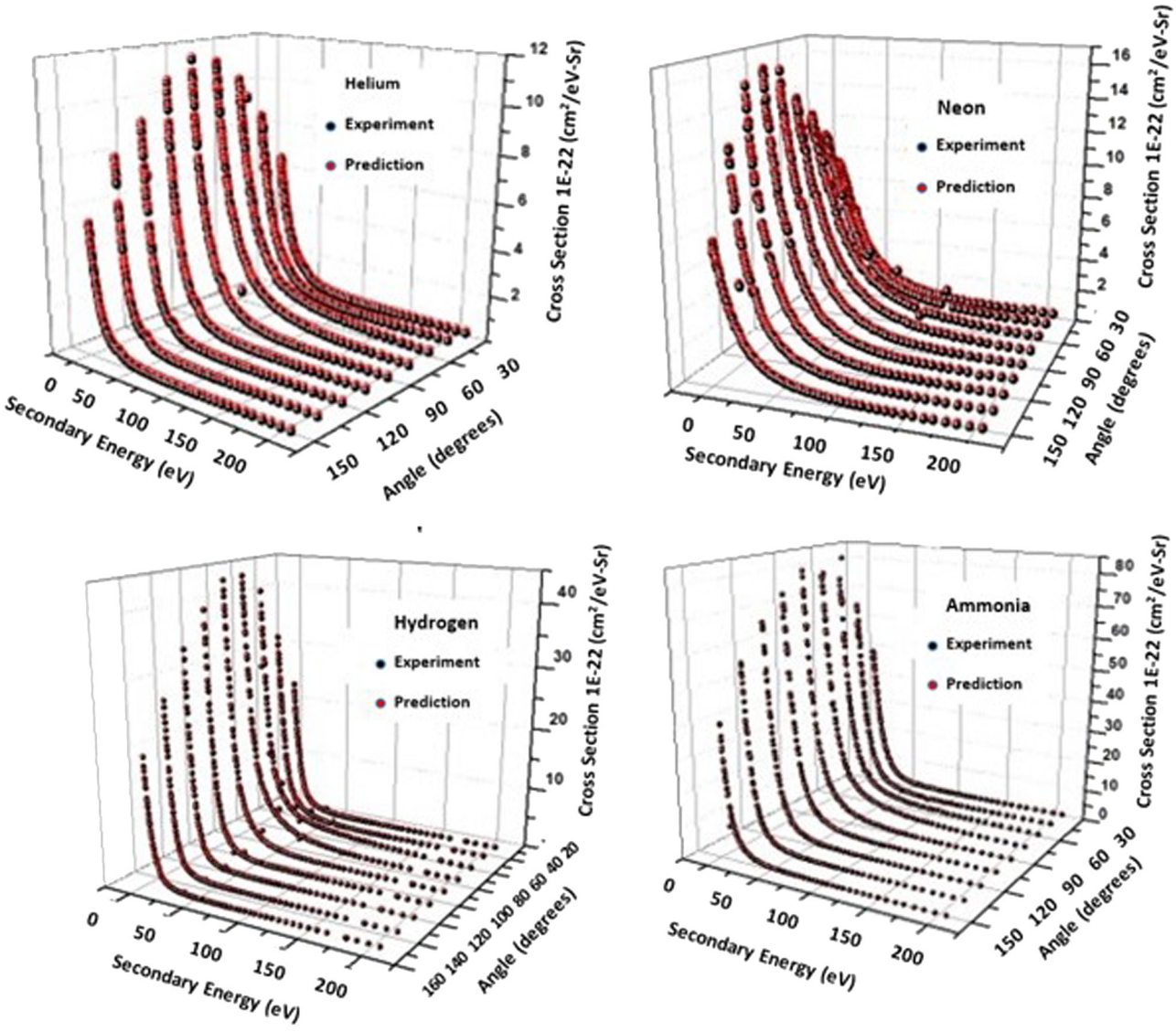

Figure 4: Differences between the experimental and predicted data for helium, neon, hydrogen, and ammonia DCS.

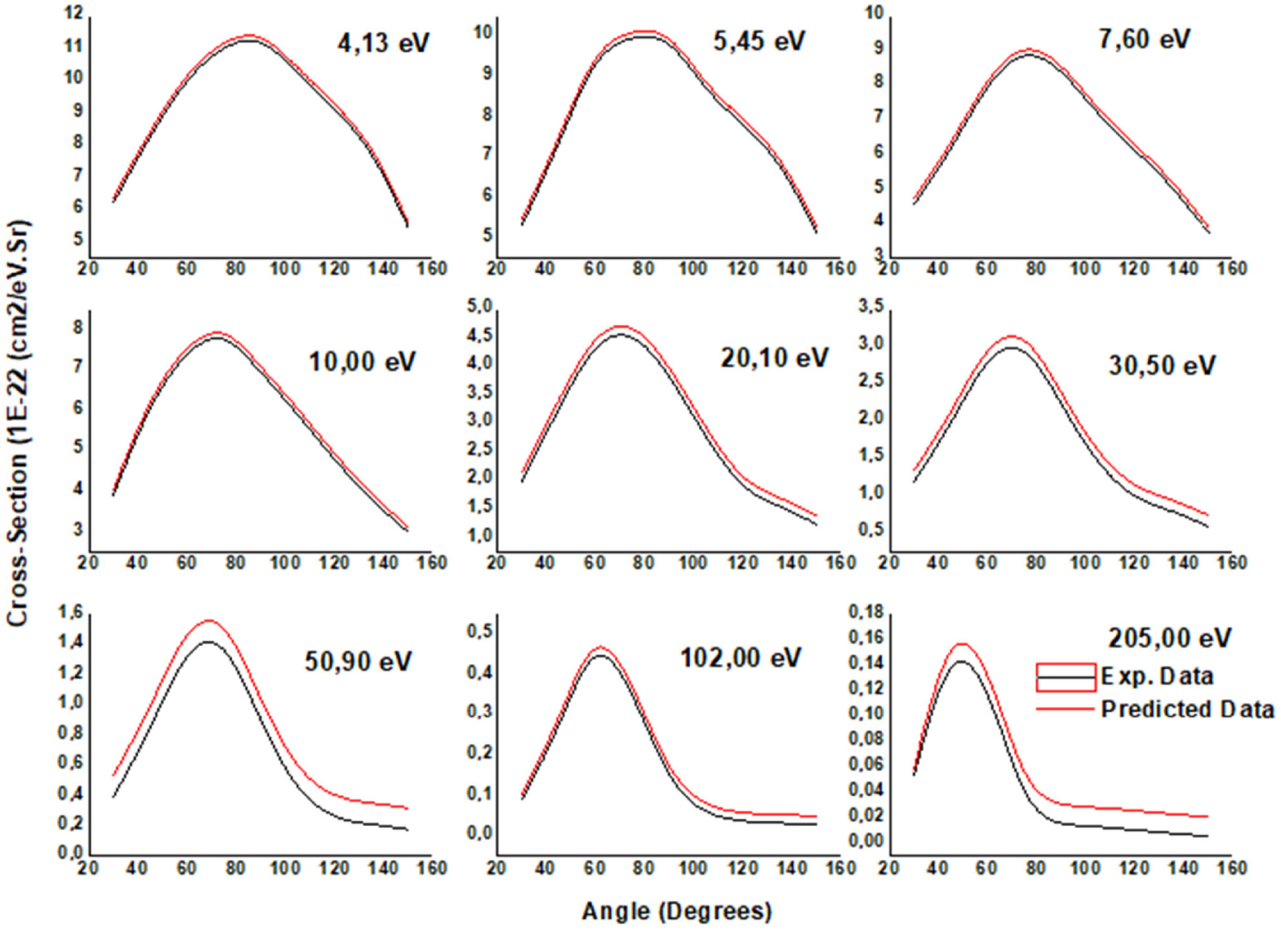

Figure 5: Comparison of experimental data with predicted DCS data for helium.

the MSE value from 29.90 to 2.11, the RMSE value from 5.46 to 1.45, the MAE value from 3.81 to 0.98, and the R^2 score from 0.83 to 0.99.

2.3 Deep learning algorithms

In the present study, the dataset was trained with different layers of deep learning models. Mobilenet and VGG architectures, which are frequently used in image classification and object recognition, are arranged according to one-dimensional input values. Table 4 shows the error metrics of these models. VGG16 and MobilenetV2 have been observed to produce acceptable results.

The deep learning models with different layers used in our study did not achieve as good results compared to the regression algorithms.

The comparison results between GBR prediction and ground truth values outlier for prediction cross-section data for He is given in Table 5. The results of the GBR showed good agreement with the DCS data. The MSE of the model was 0.69, which indicates a strong correlation between the two datasets. The results of this study demonstrate that GBR is a powerful tool for accurately predicting DCS data.

The results of ML algorithm models will depend on the type of model and training data. The differences between the experimental data and ML predictions on the training data for some atoms and molecules are presented in Figure 4. The GBR algorithm proposed in the study predicted the experimental data with a very small margin of error. The results show that the algorithm captures the underlying complexity and diversity of the data, even when it deviates from the main trend.

In Figure 5, we compare the experimental data and the predicted DCS data for helium using the GBR. A close alignment between the black line (experimental data) and the red line (predicted data using GBR) indicates a successful prediction, suggesting that the ML model has effectively learned the relationships and patterns present in the data.

3 Conclusion

ML algorithms can facilitate the analysis and prediction tasks in physics compared to traditional methods. They can process large datasets quickly and perform complex calculations, allowing researchers to focus on higher-level analysis and interpretation. Thus, the ML algorithms are an important step forward in developing accurate and reliable computational methods for prediction. In this context, we proposed a novel ML framework to predict the cross-section data. In our study, the most accurate results were tried to be obtained by using 13 different ML and 8 different deep learning algorithms and autoencoder. Within the scope of the results, the GBR algorithm reached the lowest error values compared to other algorithms. The GBR algorithm reached R^2 value of 0.99 by performing hyperparameter optimization without any processing. Low error values were also obtained with the autoencoder + KNN regression algorithm.

It is important to note that while ML algorithms can be powerful tools for prediction in physics, they should be used alongside established physics principles, theories, and models. Combining the strengths of ML with physics domain expertise can lead to valuable insights and advancements in the field of physics.

Funding information: The authors state no funding involved.

Author contributions: All authors have accepted responsibility for the entire content of this manuscript and approved its submission.

Conflict of interest: The authors state no conflict of interest.

References

[1] Mott NF. The collision between two electrons. Proceed Roy Soc A. 1930;126(801):259–67.
[2] Opal CB, Beaty EC, Peterson WK. Tables of secondary-electron-production cross-sections. At Data Nucl Data Tables. 1972;4:209–53.
[3] Oda N. Energy and angular distributions of electrons from atoms and molecules by electron impact. Radiat Res. 1975;64(1):80–95.
[4] Kurepa MV, Vuskovic L. Differential cross-sections of 100, 150 and 200 eV electrons elastically scattered in helium. J Phys B. 1975;8(12):2067–78.
[5] Kim YK. Energy distribution of secondary electrons. II. Normalization and extrapolation of experimental data. Rad Res. 1975;64(2):205–16.
[6] Jansen RHJ, de Heer FJ, Luyken HJ, van Wingerden B, Blaauw HJ. Absolute differential cross-sections for elastic scattering of electrons by helium, neon, argon and molecular nitrogen. J Phys B. 1976;9(2):185–212.
[7] Shyn TW. Angular distribution of electrons elastically scattered from gases: 2-400 eV on He. Phys Rev A. 1980;22(3):916–22.
[8] Duguet A, Lahmam-Bennani A, Lecas M, El-Marji M. A multidetection, multicoincidence spectrometer for (e,2e), and (e,3e) electron impact ionization measurements. Rev Sci Instrum. 1998;69:3524–36.
[9] Lahmam-Bennani A. Thirty years of experimental electron–electron (e,2e) coincidence studies: achievements and perspectives. J Elect Spect Rel Phen. 2002;123:365–76.
[10] Al-Hagan O, Kaiser C, Madison D, Murray AJ. Atomic and molecular signatures for charged-particle ionization. Nat Phys. 2009;5:59–63.
[11] Isik N, Dogan M, Bahceli S. Triple differential cross section measurements for the outer valence molecular orbitals (1t2) of a methane molecule at 250 eV electron impact. J Phys B: Mol Opt Phys. 2016;49:065203.
[12] Mouawad L, Hervieux PA, Dal Cappello C, El Bitar Z. Ionization of phenol by single electron impact: triple differential cross sections. J Phys B: Mol Opt Phys. 2020;53:025202.
[13] Lahmam-Bennani A, Taouil I, Duguet A, Lecas M, Avaldi L, Berakdar J. Origin of dips and peaks in the absolute fully resolved cross sections for the electron-impact double ionization of He. Phys Rev A. 1999;59:3548.
[14] Stephen K, Helm H, Mark TD. Mass spectrometric determination of partial electron impact ionization cross sections of He, Ne, Ar and Kr from threshold up to 180 eV. J Chem Phys. 1980;73:3763–78.
[15] Mirsaleh-Kohan N, Robertson WD, Compton RN. Electron ionization time-of-flight mass spectrometry: historical review and current applications. Mass Spect Rev. 2008;27:237–85.
[16] Fitch WL, Sauter AD. Calculation of relative electron impact total ionization cross sections for organic molecules. Anal Chem. 1983;55(6):832–5.
[17] Hwang W, Kim Y-K, Rudd ME. New model for electron-impact ionization cross sections of molecules. J Chem Phys. 1996;104:2956–66.
[18] Denifl S, Ptasińska S, Gstir B, Scheier P, Märk TD. Electron impact ionization of 5- and 6-chlorouracil: appearance energies. Int J Mass Spectr. 2004;232:99–105.
[19] Caleman C, Ortiz C, Marklund E, Bultmark F, Gabrysch M, Parak FG, et al. Radiation damage in biological material: Electronic properties and electron impact ionization in urea. Europhys Lett. 2009;85(1):18005.
[20] Ali E, Chakraborty HS, Madison DH. Improved theoretical calculations for electron-impact ionization of DNA analogue molecules. J Chem Phys. 2020;152:124303.
[21] Mark TD. Ionization by electron impact. Plasma Phys Control Fusion. 1992;34:2083.
[22] Becker KH, Tarnovsky V. Electron-impact ionization of atoms, molecules, ions and transient species. Plazma Sources Sci Tech. 1995;4(2):307.
[23] Kim YK, Hwang W, Weinberger NM, Ali MA, Rudd ME. Electron-impact ionization cross sections of atmospheric molecules. J Chem Phys. 1997;106:1026–33.
[24] Gluch K, Scheier P, Schustereder W, Tepnual T, Feketeova L, Mair C, et al. Cross sections and ion kinetic energies for electron impact ionization of CH4. Int J Mass Spectr. 2003;228:307–20.

[25] Goswami B, Naghma R, Antony B. Calculation of electron impact total ionization cross sections for tungsten, uranium and their oxide radicals. Int J Mass Spectrometry. 2014;372:8–12.
[26] Isik N, Yavuz M, Aksoy E, Ozer ZN, Ulu M, Sahlaoui M, et al. Comparison of experimental and theoretical double differential cross sections of CH4 at 250 eV impact energy. Acta Phy Pol A. 2015;127(4):1112–4.
[27] Builth-Williams JD, Bellm SM, Jones DB, Chaluvadi H, Madison DH, Ning CG, et al. Experimental and theoretical investigation of the triple differential cross section for electron impact ionization of pyrimidine molecules. J Chem Phys. 2012;136:024304.
[28] Milne-Brownlie DS, Foster M, Gao J, Lohmann B, Madison DH. Young-type interference in (e,2e) ionization of H_2. Phys Rev Lett. 2006;96:233201.
[29] Lee W, Nam HS, Kim YG, Kim YJ, Lee JH, Yoo H. Robust autofocusing for scanning electron microscopy based on a dual machine learning network. Sci Rep. 2011;11:20933.
[30] Tranter AD, Slatyer HJ, Hush MR, Leung AC, Everett JL, Paul KV, et al. Multiparameter optimisation of a magneto-optical trap using machine learning. Nat Comm. 2018;9:4360.
[31] Isik AH. Prediction of two-element cylindrical electrostatic lens parameters using dynamic artificial neural network. Act Phys Pol A. 2015;127:1717–21.
[32] Isik N. Determination of electron optical properties for aperture zoom lenses using an artificial neural network method. Microsc Microanal. 2016;22:458–62.
[33] Isik AH, Isik N. Time series artificial neural network approach for prediction of optical lens properties. Acta Phys Polonica A. 2016;129:514–6.
[34] Jiang J, Chen M, Fan JA. Machine neural networks for the evaluation and design of photonic devices. Nat Rev Mat. 2021;6:679–700.
[35] Mansimov E, Mahmood O, Kang S, Cho K. Molecular geometry prediction using a machine generative graph neural network. Sci Rep. 2019;9:20381.
[36] Wei JN, Belanger D, Adams RP, Sculley D. Rapid prediction of electron−ionization mass spectrometry using neural networks. ACS Cent Sci. 2019;5:700–8.
[37] Stevenson C, Pérez-Ríos J. Genetic based fitting techniques for high precision potential energy curves of diatomic molecules. J Phys B: Mol Opt Phys. 2019;52:105002.
[38] Zhong L. Fast prediction of electron-impact ionization cross-sections of large molecules *via* machine learning. J Appl Phys. 2019;125(18):183302.
[39] Liu X, Meijer G, Pérez-Ríos J. A data-driven approach to determine dipole moments of diatomic molecules. Phys Chem Chem Phys. 2020;22:24191–200.
[40] Liu X, Truppe S, Meijer G, Pérez-Ríos J. The diatomic molecular spectroscopy database. Cheminform. 2020;12:31.
[41] Nam J, Yong H, Hwang J, Choi J. Training an artificial neural network for recognizing electron collision patterns. Phys Lett A. 2021;127005.
[42] Cretu MT, Pérez-Ríos J. Predicting second virial coefficients of organic and inorganic compounds using Gaussian process regression. Phys Chem Chem Phys. 2021;23:2891.
[43] Du YL, Pablos D, Tywoniuk K. Machine learning jet modifications in heavy-ion collisions. J High En Phys. 2021;206:1–49.
[44] Albertsson K. Machine learning in high-energy physics: Displaced event detection and developments in ROOT/TMVA. Doctoral thesis. Sweden: Lulea University of Technology; 2021.
[45] Canudas NV, Gómez MC, Ribé EG, Vilasis-Cardona X. Use of deep learning to improve the computational complexity of reconstruction algorithms in high energy physics. Appl Sci. 2021;11:11467.
[46] Ambrosino F, Sabbarese C, Roca V, Giudicepietro F, Chiodini G. Analysis of 7-years Radon time series at Campi Flegrei area (Naples, Italy) using artificial neural network method. App Rad Isotopes. 2020;163:109239.
[47] Grojean C, Paul A, Qian Z, Strümke I. Lessons on interpretable machine learning from particle physics. Nat Rev Phys. 2022;4:284–6.
[48] Livingstone DJ, Manallack DT, Tetko IV. Data modelling with neural networks: Advantages and limitations. J Comput Mol Des. 1997;11:135–42.
[49] Bonaccorso G. Machine learning algorithms: A reference guide to popular algorithms for data science and machine learning. Birmingham: Packt Publishing; 2017.
[50] Alpaydin E. Machine learning. MIT Press: Massachusetts Institute of Technology; 2016.