Cumhuriyet Sci. J., 46(2) (2025) 233-239
DOI: https://doi.org/10.17776/csj.1576821

Cumhuriyet Science Journal

| csj.cumhuriyet.edu.tr | Founded: 2002 ISSN: 2587-2680 e-ISSN: 2587-246X Publisher: Sivas Cumhuriyet University

Molecular pKa Prediction with Deep Learning and Chemical Fingerprints

Fatih Mehmet Avcu [1,a,*]

[1] *Department of Informatics, İnönü University, 44280, Malatya, Türkiye*

**Corresponding author*

Research Article

History
Received: 31/10/2024
Accepted: 28/04/2025

ABSTRACT

Today, drug discovery and design, the determination of molecular properties, in particular the determination of a molecule's pKa value, is essential for understanding and optimising the biological activity of drugs. In this context, in addition to traditional chemical methods, artificial intelligence techniques such as machine learning and deep learning are increasingly used to predict molecular properties and drug design processes. In this paper, we present an approach that investigates the effect of molecular properties on pKa prediction and implements this prediction using a deep learning model. The model considers molecular weight together with chemical fingerprinting methods such as Morgan fingerprinting to represent molecular structures. The dataset used in this study contains 2093 molecular data points obtained from PubChem. The method presented in the paper predicts the pKa values of many molecules with 96.66% accuracy. This can save time and money in the drug discovery, design process, and provide valuable guidance for experimental studies. The paper also presents a comprehensive analysis of the training process, accuracy metrics and performance of the deep learning model. Finally, this paper presents research that evaluates the impact of molecular features on pKa prediction and demonstrates the success of the deep learning model in these predictions.

Keywords: SMILES, pKa, Machine Learning, Deep Learning.

[a] *fatih.avcu@inonu.edu.tr* *https://orcid.org/0000-0002-1973-7745*

Introduction

The pKa values of molecules determine many important properties such as the direction and rate of chemical reactions, solubility, biological activity and environmental behaviour. The determination of pKa values by conventional methods requires laboratory experiments. Therefore, rapid and accurate estimation of pKa values is of great importance in chemistry and biochemistry.

In recent years, the application of artificial intelligence and machine learning techniques to chemistry has led to significant advances in the prediction of molecular properties. These techniques can model complex properties of molecular structures by learning from large data sets and predict them with high accuracy. In particular, deep learning models have the potential to make successful predictions by processing the structural information of molecules.

In this study, a deep learning model was developed to predict the pKa values of molecules using Simplified Molecular Input Line Entry System(SMILES) codes. SMILES is a common notation system that represents the structures of molecules in text format and is widely used in chemical information processing. The model developed uses Morgan fingerprinting to represent molecular structures, taking into account molecular weight.

This paper details the steps taken to develop the model, the data sets used, the training and validation processes, and the performance analyses of the model.

Related Works

Estimating the pKa of molecules is a topic that has long been studied in chemistry and biochemistry. Traditionally, pKa values are determined by laboratory methods such as titration experiments. These methods provide high accuracy but are time consuming and costly. Therefore, in recent years there has been a great deal of interest in computational estimation methods.

Quantum chemistry and molecular dynamics approaches

Quantum chemical calculations and molecular dynamics simulations attempt to predict pKa values by calculating molecular structures and energy levels. Gao et al (2009) successfully predicted the pKa values of several organic molecules using quantum mechanical and molecular mechanical (QM/MM) approaches [1]. Ho and Coote studied the prediction of acidity in the gas and solution phases from the first principles [2]. Cramer and Truhlar developed methods for transition metal chemistry using density functional theory [3].

Machine learning approaches

Machine learning techniques, especially deep learning models, have revolutionised the prediction of molecular properties. Xu et al. predicted pKa values on a large dataset using machine learning algorithms and achieved high accuracy [4]. Wang et al. improved the accuracy of pKa predictions by combining deep learning models and chemical fingerprinting methods [5]. Mayr et al. investigated toxicity prediction using deep learning and developed a model called DeepTox [6]. Ramsundar et al.

presented a comprehensive review of the use of deep learning in life sciences [7]. Feinberg et al. achieved significant success in molecular feature prediction using a model called PotentialNet [8]. Gilmer et al. developed highly accurate models for quantum chemistry using neural message passing techniques [9]. Wu et al. evaluated molecular machine learning models using a benchmark called MoleculeNet [10]. Rupp et al. quickly and accurately modelled molecular atomisation energies using machine learning [11]. Faber et al. showed that molecular machine learning models have lower prediction errors than hybrid DFT errors [12]. Schütt et al. developed a convolutional neural network with a continuous filter called SchNet for modelling quantum interactions [13]. De Cao and Kipf introduced a generative model called MolGAN for small molecular graphs [14]. Kearnes et al. managed to go beyond fingerprints using molecular graph convolutions [15]. Winter et al. learned continuous and data-driven molecular descriptors by transforming equivalent chemical representations [16]. Altae-Tran et al. used one-shot learning techniques for low-data drug discovery [17]. Ragoza et al. achieved significant success in protein-ligand scoring using convolutional neural networks [18].

SMILES and chemical fingerprinting methods

SMILES codes are a common notation system that represents the structures of molecules in a compact and understandable form. Chemical fingerprinting methods are widely used to numerically represent the structural properties of molecules. Rogers and Hahn provided a detailed encoding of molecular structures using the Morgan fingerprinting method and showed that this method provides important input for machine learning models [19].

In this study, a deep learning model was developed to predict the pKa values of molecules using SMILES codes and chemical fingerprinting methods. The developed model aims to increase the accuracy of pKa predictions by taking into account traditional molecular properties. Considering previous studies in the literature, it seems that this approach provides a significant improvement in pKa predictions and can be a valuable tool for chemical research.

Materials and Methods

Data Collection

The data used in this study were obtained from the PubChem database. PubChem is an open-access resource providing comprehensive information on chemical compounds, biomolecules and biological activities. The PubChempy library developed for this purpose was used.

SMILES and its Implementation

SMILES is a naming system designed to represent the structure of chemical molecules in text format. This representation consists of a sequence of characters without spaces. SMILES has found wide application in chemical information processing by expressing the atomic and bonding structures of molecules in human-readable and writable strings. The SMILES format is widely used in chemical databases due to its ability to efficiently encode both simple and complex molecules. This convenience of SMILES allows chemical data to be efficiently stored, searched and compared [20]. Furthermore, the flexibility and ease of use of the SMILES format have led to its preference in many chemical data processing software and databases [21]. These features make SMILES an important tool in the field of chemical information systems.

Table 1 compares the the International Union of Pure and Applied Chemistry (IUPAC) names of various molecules with their SMILES representations. For example, the SMILES representation of the water molecule called 'oxidane' is 'O'. The more complex molecule 6-(hydroxymethyl)oxane-2,3,4,5-tetrol is represented as 'C(C1C(C(C(C(O1)O)O)O)O)O)O'. This table clearly demonstrates the ability of the SMILES format to represent both simple and complex molecules.

Table 1. IUPAC names of molecules, SMILES representations

IUPAC Names	SMILES Representations
Oxidane	O
Ethanol	CCO
Acetic acid	CC(=O)O
Benzene	c1ccccc1
6-(hydroxymethyl)oxane-2,3,4,5-tetrol	C(C1C(C(C(C(O1)O)O)O)O)O)O
2-acetyloxybenzoic acid	CC(=O)Oc1ccccc1C(=O)O
Metan	C
Propan	CCC

Chemical Fingerprinting Methods

The Morgan Fingerprint is a molecular descriptor widely used in chemoinformatics. It represents the structural properties of molecules in a numerical format that is ideal for comparing molecular similarities and predicting molecular properties in machine learning models. Morgan FingerPrint takes into account the chemical environment within a certain radius from the atoms of the molecules and encodes this environment as a unique bit sequence. This method, commonly known as Extended-Connectivity FingerPrints(ECFP), captures the topological properties of molecules, effectively representing the environmental information of chemical bonds and atoms. For example, the Morgan fingerprint of a molecule is calculated based on the chemical bonding pattern of atoms within a given radius, and this information is uniquely encoded in a bit string. In this way, Morgan fingerprints can be used quickly and efficiently for molecular similarity searches in large chemical databases and for predicting associated biological activities.

Table 2 compares the representation of several molecules in the SMILES format with the 64-bit Morgan FingerPrint representation of these molecules. The table shows how Morgan FingerPrint encodes chemical structures into a numerical format and captures the topological properties of molecules. Örneğin,

'OC=1(N(N=C(C=1)C2(=CC=CC=C2))C)' The molecule represented by the SMILES notation is encoded as a bit string in the Morgan FingerPrint format '1010010010000000010010101000000111001001100110010011010001100000'. This table helps to understand the use of Morgan FingerPrint for calculating molecular similarities and for fast searching in chemical databases.

Table 2. SMILES and 64-bit Morgan FingerPrint Representations of Molecules

SMILES Demonstration	64-bit Morgan FingerPrint
OC=1(N(N=C(C=1)C2(=CC=CC=C2))C)	1010010010000000010010101000000111001001100110010011010001100000
S(=O)(=O)(NC1(=NC(=CC=N1)C))C2(=CC=C(N)C=C2)	1001100001111000110101100000100100100100000001010010101001
ON=C1(C(=NO)CCCCC1)	0011100000100000001001010000001000001001100010000000001000000000
O=[N+]([O])C1(=CC(=CC=C1)[CH][N+]3(=CC=C(C(=O)CC2(=CC=CC=C2))C=C3))	1001011000110011111010101001100010100101001011101110100000100100100
O=C(O)C=1(OC=2(C=CC=CC=2(C=1OC)))	1000110011100100110001100010001001001101100001011000000100000101

Deep Learning Model

Deep learning, a subset of machine learning, is a technique that uses artificial neural networks to learn and make decisions from data. When supported by large data sets and high computing power, this method is capable of solving complex problems and achieving high accuracy rates. Deep learning models use artificial neural networks consisting of many layers and learn complex relationships in the data through these layers [22].

The basic building block of deep learning is the artificial neural network. Artificial neural networks are inspired by biological neural networks and consist of interconnected artificial neurons (Fig. 1). Each neuron receives inputs multiplied by a given weight, processes them through an activation function, and produces an output. The network is organised in layers; the input layer receives the data, hidden layers process the data and the output layer produces the final prediction or classification result. This multi-layer structure forms 'deep' neural networks, and deeper networks are generally capable of learning more complex and abstract features [23].

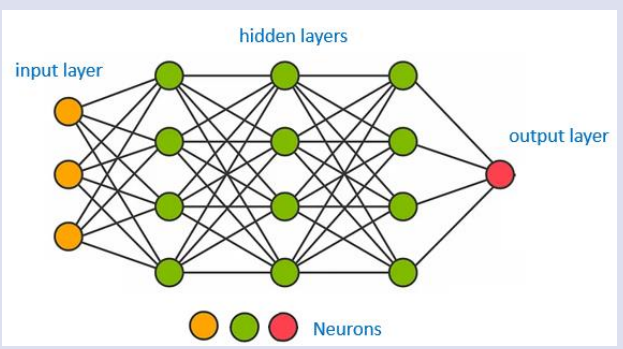

Figure 1. Schematic representation of Deep Learning

Deep learning models are trained on large datasets. The training process involves optimising the parameters (weights) of the model. This is done by minimising the difference (loss) between the model's predicted values and the actual values. Optimisation algorithms such as Stochastic Gradient Descent (SGD) or Adam are used to minimise this loss [24]. In each training cycle, the model learns from the data and updates its parameters.

Deep learning becomes more effective as the amount of data and computing power increases. Today, thanks to GPUs and specialised hardware (e.g. TPUs), deep learning models can be trained quickly and efficiently on large datasets [25]. The development of this technology is enabling artificial intelligence to achieve results close to or better than human performance in several areas.

Metrics Used

Mean Absolute Error (MAE) is a performance metric that measures the average absolute difference between a model's predicted and actual values. The MAE is used to assess the accuracy of predictions and is calculated as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \quad (1)$$

In Eq.1, $\hat{y}_i$ is the predicted value, y_i is the true value and n is the total number of data points. The MAE shows the average error of the model's predictions and is easy to interpret. Low MAE values indicate that the model is predicting with high accuracy.

Mean Squared Error (MSE) is a performance metric that measures the average of the squares of the differences between a model's predicted and actual values. The MSE is used to assess the amount of error in the model's predictions and the magnitude of these errors and is calculated as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \quad (2)$$

In Eq.2, $\hat{y}_i$ is the predicted value, y_i is the true value and n is the total number of data points. The MSE squares and magnifies errors, so that large errors are penalised

more. Low MSE values indicate that the model is predicting with high accuracy.

Software and Algorithm

Computations were performed on an Intel 8700 processor computer running Ubuntu 24.04 LTS Linux with kernel version 5.4.0-12.15-generic. All computations were performed using Python 3.12.4, Scikit-learn 1.4.2, Tensorflow 2.17.0 and Keras 3.4.1. A schematic representation of the developed software is shown in Figure 2.

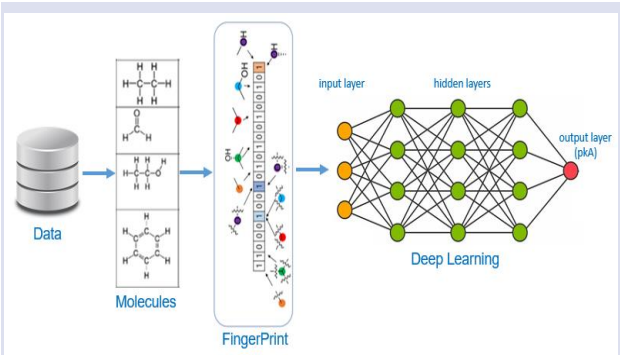

Figure 2. Schematic representation of algorithm

Analysis and Results

In machine learning, all data features must be on the same scale [26]. Differences between the original features in the dataset can cause problems for many machine learning models. Variables measured at different scales do not contribute equally to the model fit and learning function, which can lead to biased results. For example, if a feature in the dataset is numerically large, it will become dominant. To avoid this, I scale the data using standardisation or z-score normalisation. To do this, I use the StandardScaler function in the Sklearn.Preprocessing library in Python. StandardScaler sets the mean of each column in the data set to 0 and the standard deviation to 1.

To evaluate the performance of the model during the classification process, 30% of the dataset was randomly selected as test data. The remaining 70% was used to train the model. This split of the dataset allows a more general and reliable evaluation of the model's performance. This was done using the 'train_test_split' function in the 'sklearn.model_selection' library in the Python programming language. This function allows the model to be accurately evaluated on both training and test data, while randomly splitting the data set. In this way, the overall validity and generalisability of the model is tested.

In order to prevent overfitting during the development of the model, the "Dropout" technique was applied to each hidden layer. This technique increases the generalisation capability of the model and reduces the risk of overfitting by randomly disabling neurons at certain rates during the learning process of the model[27-28]. In this study, 30% of the neurons were randomly disabled for each hidden layer.

In the optimisation process, the Adaptive Moment Estimation (Adam) algorithm was used, which is widely preferred and provides effective results[29]. By adapting the learning rate for each parameter separately, Adam provides successful results, especially in deep learning models[30]. The deep learning model was trained for a total of 750 epochs; this number of epochs was chosen to allow the model to learn and generalise sufficiently. MAE and MSE functions were used to evaluate and optimise the performance of the model. In understanding the error distribution, MAE treats all errors equally, while MSE penalises large errors more. Using these two metrics together, one can evaluate both the overall error size (MAE) and the sensitivity of the model to large errors (MSE). In performance balancing, using MAE and MSE together ensures that the model performs well against both small and large errors. For example, if a model avoids making very large errors, this may result in a large improvement in MSE, but not necessarily the same improvement in MAE. A low MAE and MSE function is an indication that the model is being better optimised by making accurate predictions. Therefore, minimising the loss function during the training process is critical to the overall performance of the model.

The model created in this study consists of 5 layers in total. These consist of 1 input, 1 output and 3 hidden layers. The structure used 128 input neurons and a hidden layer of 256 neurons. The rectified linear unit (ReLU) function was preferred as the activation function of the input and hidden layers. A linear function was used to activate the output layer. The output layer of the pKa prediction system was configured as 1 neuron.

Table 3. Performance Metrics across Different Hidden Layers and Epochs

HL	Epoch	MAE	MSE	Accuracy
1	100	0.0317	0.0092	0.8559
1	250	0.0229	0.0092	0.7889
1	500	0.0187	0.0093	0.7978
1	750	0.0179	0.0092	0.748
1	1000	0.0179	0.0094	0.7684
2	100	0.0394	0.0099	0.9508
2	250	0.0239	0.0086	0.9611
2	500	0.0175	0.0081	0.9303
2	750	0.0177	0.008	0.9413
2	1000	0.0177	0.0082	0.9549
3	100	0.0353	0.009	0.9598
3	250	0.021	0.0079	0.9545
3	500	0.0172	0.0076	0.9659
3	750	0.0166	0.0076	0.9666
3	1000	0.0178	0.0079	0.9666

The main objective in choosing the different hidden layer configurations listed in Table 3 is to balance the complexity of the model with the overall accuracy. Increasing the number of hidden layers allows the model to learn more parameters and better represent complex relationships in the dataset, while at the same time it may

increase the risk of overfitting. Although fewer hidden layers allow the model to learn faster and have a higher generalisation capacity, the accuracy level may be lower for more complex data. The experiments conducted in the study showed that the model with three hidden layers gave the best results in terms of both accuracy and error rates. For example, this structure showed the highest performance with an accuracy of 96.66% between 750 and 1000 epochs, while at the same time increasing the capacity to learn complex relationships with low error rates (MAE and MSE). After 750 epochs, the performance stabilised, indicating the optimal learning limit for this structure and avoiding the risks of adding more layers. These results show that deep structures offer better generalisation capacity in pKa estimation and allow for a balanced optimisation of small and large errors. These results are more clearly visualised in Figures 3, 4 and 5.

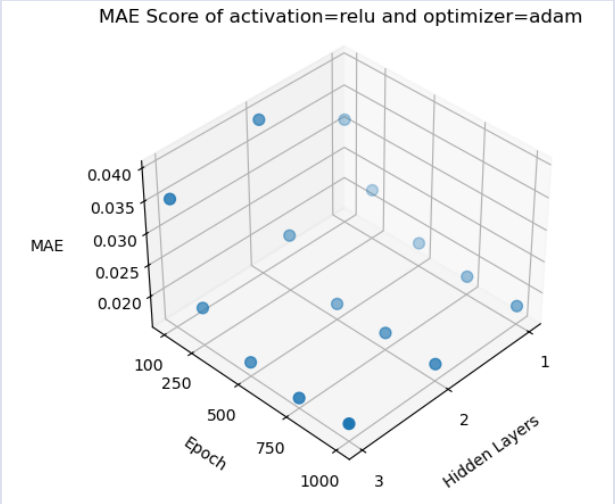

Figure 3. MAE Results: Change with Number of Layers and Epoch

Figure 3 shows the (MAE) performance of the model for different number of layers and epoch values. In this figure, the sensitivity of the MAE, which reflects the average magnitude of the model's prediction errors, to the model's configuration parameters is examined. It is clear from the figure that adding more hidden layers generally leads to lower MAE values. In particular, the three-layer model provided the lowest MAE values at all epoch values, indicating that the model can make more accurate predictions with a deeper structure. This indicates the capacity of deep learning models to learn more complex relationships in the data. With the increase in the number of epochs, a significant decrease in MAE values was observed at first. However, after 750 epochs, the improvement in MAE values came to a halt and fluctuations occurred from time to time. This trend indicates that the learning reaches a saturation point when the model is trained further and additional training processes do not significantly reduce the error. Furthermore, no signs of overlearning were observed up to 1000 epochs, indicating that the generalization capacity of the model is high.

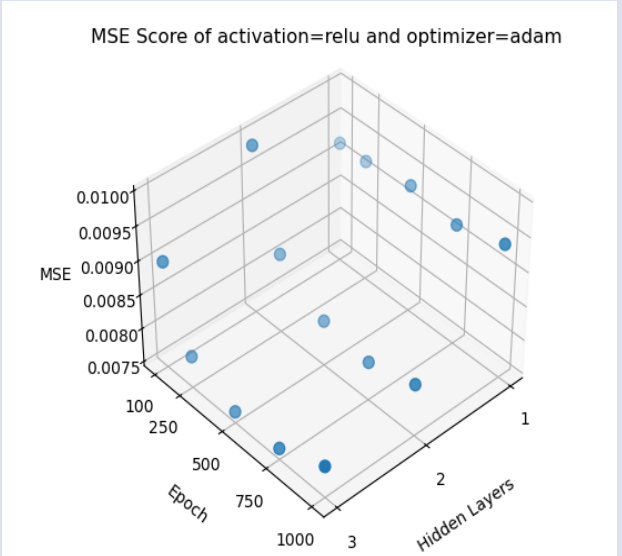

Figure 4. MSE Results: Change with Number of Layers and Epoch

Figure 4. shows the mean square error (MSE) performance of the model for different number of layers and epoch values. This graph examines the sensitivity of the model to the squared prediction errors and its dependence on the reconstruction parameters. It is observed that there is a significant decrease in the MSE values with increasing number of layers. In particular, the three-layer model provides the lowest MSE values at all epoch levels, proving that prediction accuracy improves with more complex structures. While the MSE initially decreases rapidly with the increase in the number of epochs, this decrease slows down after approximately 750 epochs and the values start to stabilize. This suggests that the model reaches a saturation point during the training process and the performance does not improve significantly with additional training. Furthermore, the absence of signs of overlearning up to 1000 epochs demonstrates the robustness of both the generalization ability and the learning process of the model.

Figure 3 and Figure 4 show an overall improvement in both MAE and MSE values as the number of layers increases, indicating that deeper models provide better learning. With a single layer structure, the model achieves 76.84% accuracy and 0.00179 MAE in 1000 epochs with 0.00178 MAE. Furthermore, the MSE values ranged from 0.00076 to 0.00094, indicating that the overall performance of the model is high.

Although increasing the number of epochs will initially improve performance, the improvement will plateau or slightly deteriorate in the 750-1000 epoch range. The epoch value is not increased further as the model may overfit after a certain point with more training. The agreement between MSE and MAE shows that the model improves similarly for small and large errors.

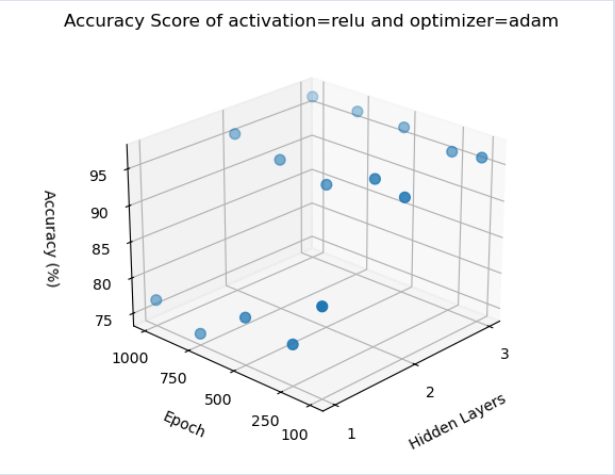

Figure 5. Accuracy Results: Comparison of Model Performance

Figure 5. shows that the three-layer model performs best in the range of 750-1000 epochs and reaches 96.66% accuracy in pKa predictions. This proves that the model can be successfully trained and predicted with both low error rates (MAE and MSE) and high accuracy. The superior performance of the three-layer model points to the capacity of deep learning models to learn complex relationships. A deeper structure allowed the model to learn the fine details in the data, producing more consistent and accurate results. In particular, both MAE and MSE values are low, indicating that the model improves small and large prediction errors similarly. It is observed that increasing the number of epochs initially improves the performance, but the improvement slows down after 750 epochs and stabilizes up to 1000 epochs. This suggests that the model reaches a saturation point at this point and additional training runs do not significantly reduce the error rates. Furthermore, the absence of signs of overfitting (overlearning) in the model proves that the model has a good generalization capacity in general and that this configuration is a reliable forecasting tool.

Conclusions

This study has shown that molecular pKa values can be predicted with high accuracy using a deep learning model. The model was trained on a dataset of 2093 molecules and achieved a prediction accuracy of 96.66%. The results show that deep learning methods are an effective tool for predicting chemical properties. In particular, careful selection and modelling of molecular features play a critical role in improving the accuracy of pKa predictions.

The results of the study can contribute to a wide range of practical applications in chemistry and biochemistry. For example, in the drug discovery and design process, rapid and accurate estimation of pKa values can be an important step in optimising the biological activities of new drug molecules. These methods can also be used in environmental chemistry to model processes such as the dispersion and biodegradability of pollutants.

For future studies, the effect of different combinations of molecular features on deep learning models can be investigated in more detail. In addition, the generalisation ability of the model can be tested and improved by using larger and more diverse data sets. Such improvements can increase the usability of the model not only in academic research, but also in industrial applications.

Conflicts of interest

There are no conflicts of interest in this work.

References

[1] Gao J., Truhlar D.G., Quantum mechanical methods for enzyme kinetics, *Annu Rev Phys Chem.,* 53 (2002) 467-505.

[2] Ho J., Coote M.L., First-principles prediction of acidities in the gas and solution phase, *WIREs Comput Mol Sci.* 1(5) (2011) 649-60.

[3] Cramer C.J., Truhlar D.G., Density functional theory for transition metals and transition metal chemistry, *Phys Chem Chem Phys.* 11(46) (2009) 10757-10816.

[4] Xu Y., Dai Z., Chen F., Gao S., Pei J., Lai L., Deep learning for drug-induced liver injury, *J Chem Inf Model.* 55(10) (2015) 2085-2093.

[5] Wang S., Guo Y., Wang Y., Sun H., Huang J., SMILES-BERT: Large scale unsupervised pre-training for molecular property prediction. *Proc 10th ACM Int Conf Bioinformatics, Comput Biol Health Inform.,* (2019) 429-436.

[6] Mayr A., Klambauer G., Unterthiner T., Hochreiter S., DeepTox: Toxicity prediction using deep learning, *Front Environ Sci.,* (2016)

[7] Ramsundar B., Eastman P., Walters P., Pande V., Deep learning for the life sciences, Sebastopol (CA): O'Reilly Media, (2015).

[8] Feinberg E.N., Sur D., Wu Z., Husic B.E., Mai H., Li Y., PotentialNet for molecular property prediction, *ACS Cent Sci.,* 4(11) (2018) 1520-1530.

[9] Gilmer J., Schoenholz S.S., Riley P.F., Vinyals O., Dahl G.E., Neural message passing for quantum chemistry. In: Proceedings of the 34th International Conference on Machine Learning, (2017) 1263-1272.

[10] Wu Z., Ramsundar B., Feinberg E.N., Gomes J., Geniesse C., Pappu A.S., MoleculeNet: A benchmark for molecular machine learning, *Chem Sci.,* 9 (2018) 513-530.

[11] Rupp M., Tkatchenko A., Müller K.R., von Lilienfeld O.A., Fast and accurate modeling of molecular atomization energies with machine learning, *Phys Rev Lett.,* 108(5) (2012) 058301.

[12] Faber F.A., Hutchison L., Huang B., von Lilienfeld O.A., Baitz G.J., Prediction errors of molecular machine learning models lower than hybrid DFT error, *J. Chem. Theory Comput.,* 13(11) (2017) 5255-5264.

[13] Schütt K.T., Kindermans P.J., Sauceda H.E., Chmiela S., Tkatchenko A., Müller K.R., SchNet: A continuous-filter convolutional neural network for modeling quantum interactions. In: Proceedings of the 31st Conference on Neural Information Processing Systems, (2017) 991-1001.

[14] De Cao N., Kipf T., MolGAN: An implicit generative model for small molecular graphs. *arXiv Preprint arXiv:1805.11973*, (2018).

[15] Kearnes S., McCloskey K., Berndl M., Pande V., Riley P., Molecular graph convolutions: Moving beyond fingerprints, *J. Comput. Aided Mo.l Des.,* 30 (2016) 595-608.

[16] Winter R., Montanari F., Noé F., Clevert D.A., Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations, *Chem. Sci.,* 10(6) (2019) 1692-701.

[17] Altae-Tran H., Ramsundar B., Pappu A.S., Pande V., Low data drug discovery with one-shot learning, *ACS Cent. Sci.,* 3(4) (2017) 283-93.

[18] Ragoza M., Hochuli J., Idrobo E., Sunseri J., Koes D.R., Protein-ligand scoring with convolutional neural networks, *J. Chem. Inf. Model.,* 57(4) (2017) 942-957.

[19] Rogers D., Hahn M., Extended-connectivity fingerprints, *J. Chem. Inf Model,* 50(5) (2010) 742-54.

[20] Weininger D., SMILES, a chemical and information system. 1. Introduction to methodology and encoding rules, *J. Chem. Inf. Model,* 28(1) (1988) 31-6.

[21] Daylight Chemical Information Systems. SMILES: Simplified molecular input line entry system. Available at: https://www.daylight.com/smiles/. Retrieved October 2, 2023.

[22] LeCun Y., Bengio Y., Hinton G., Deep learning. *Nature,* 521(7553) (2015) 436-44.

[23] Goodfellow I., Bengio Y., Courville A., Deep learning, Cambridge (MA): MIT Press, (2016).

[24] Avcu F.M., Clustering honey samples with unsupervised machine learning methods using FTIR data, *An. Acad. Bras. Cienc.,* 96(1) (2024).

[25] Jouppi N.P., Young C., Patil N., Patterson D., Agrawal G., Bajwa R., In-datacenter performance analysis of a tensor processing unit, In: Proceedings of the 44th Annual International Symposium on Computer Architecture, (2017) 1-12.

[26] Jain A.K., Murty M.N., Flynn P.J., Data clustering: a review. *ACM Comput Surv.,* 31(3) (1999) 264-323.

[27] Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., Dropout: a simple way to prevent neural networks from overfitting, *J. Mach. Learn. Res*., 15 (2014) 1929–1958.

[28] Zaremba W., Recurrent neural network regularization. *arXiv preprint arXiv:*1409.2329 (2014).

[29] Ruder S., An overview of gradient descent optimization algorithms, *arXiv Preprint arXiv:1609.04747* (2017).

[30] Karakaplan M., Avcu F.M., Classification of some chemical drugs by genetic algorithm and deep neural network hybrid method, *Concurr Comput Pract Exp.,* 33(13) (2021).