Preparation of Simplified Molecular Input Line Entry System Notation Datasets for use in Convolutional Neural Networks

Sandi Baressi Šegota
Department of Automation and Electronics[1]
Faculty of Engineering – University of Rijeka
Rijeka, Croatia
sbaressisegota@riteh.hr

Nikola Anđelić
Department of Automation and Electronics
Faculty of Engineering – University of Rijeka
Rijeka, Croatia
nandelic@riteh.hr

Ivan Lorencin
Department of Automation and Electronics
Faculty of Engineering – University of Rijeka
Rijeka, Croatia
ilorencin@riteh.hr

Jelena Musulin
Department of Automation and Electronics
Faculty of Engineering – University of Rijeka
Rijeka, Croatia
jmusulin@riteh.hr

Daniel Štifanić
Department of Automation and Electronics
Faculty of Engineering – University of Rijeka
Rijeka, Croatia
dstifanic@riteh.hr

Zlatan Car
Department of Automation and Electronics
Faculty of Engineering – University of Rijeka
Rijeka, Croatia
car@riteh.hr

***Abstract*— Simplified Molecular Input Line Entry System (SMILES) is a type of chemical notation. The SMILES format allows the representation of chemical structures in a shape easily readable by computer programs. This allows many techniques, such as Artificial Neural Networks (ANNs) to be applied on the SMILES formatted data. One of the highest-performing ANN types is the Convolutional Neural Networks (CNNs), designed to work on images or matrix-shaped data. In this paper, the authors will present the preparation of the SMILES dataset for use by CNNs. The paper will start with a brief description of the SMILES format, followed by the explanation of the dataset transformation into an NPY matrix-based format, with an example of utilization via the application of popular CNN architectures on a transformed dataset. The proposed architecture achieves satisfactory results (AUC=0.92), with the transformation algorithm speed also proving satisfactory (0.08 seconds per data point)**

Keywords—artificial intelligence, convolutional neural networks, data processing and transformation, machine learning, SMILES

I. INTRODUCTION

Artificial Intelligence (AI) is a powerful tool, that is used in a variety of fields today – medicine [1,2], robotics [3,4], epidemiology [5, 6], economics [7], and others [8]. Many authors provide examples of the use of AI algorithms, mainly machine learning (ML) algorithms, such as ANNs in biochemistry and bioengineering. For example, Gruson et al. (2020) [9] provide examples of data science, AI, and ML applications in laboratory medicine, noting the value of positive regulation. Heo et al. (2021) [10] propose the utilization of AI tools for physics-based protein structure refinement in the era of AI, where authors propose multiple techniques for the application of AI algorithms for the task of protein structure refinement. Sisman and Basok (2020) [11] discuss the needs and possibilities with the application of digitalization and AI in laboratory medicine. Peña - Guerrero et al. (2021) [12] discuss the growing applications of ML, AI, and data science for drug design and neglected diseases. Bai-Shuoyan et al. (2021) [13] discuss the elements of the 3D drug design of protein targets. The authors propose MolAICal, a software tool that achieves results in designing protein target drugs using AI. Another discussion on drug design is provided by Jiménez-Luna et al. (2020) [14], where the authors discuss possible implementations and current state-of-the-art research in the field.

The application of ML techniques to problems in bioengineering and biochemistry requires data formatting. ML algorithms learn from the data provided to them, and as such require data to be computer-readable [15]. A computer-readable molecule describer format does exist, in the shape of SMILES [16]. There is several previous research focusing on the use of SMILES-based datasets for AI-based applications. Pinheiro et al. 2020 [17] apply the transformations of the molecules in the QM9 dataset [18] into the free coordinate descriptors and using an ML algorithm – Feed Forward Neural Network (FNN) predict nine different molecular properties with high precision. Van Deursen et al. [19] (2020) propose a SMILES explorer which uses autodidactic generative examination networks, showing high performance in the task. Yang et al. (2017) [20] demonstrate a Python library for molecule generation, which uses a SMILES trained Recurrent Neural Network (RNN) in the rollout procedure. There is a lack of papers discussing the application of AI on SMILES-based datasets, especially in the case of CNNs. CNNs are algorithms with high performance in classification and regression [21], commonly applied on image data due to the convolution procedure being applied, which works well on matrix-shaped data [22]. To apply CNNs the SMILES data needs to be appropriately formatted as a matrix. This paper first presents the transformation process of the SMILES dataset into a matrix-shaped NPY dataset, using a publicly available TOX21 dataset [23]. Then, the transformation is tested through the application of a

This research has been (partly) supported by the CEEPUS network CIII-HR-0108, European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS), project CEKOM under the grant KK.01.2.2.03.0004, CEI project "COVIDAi" (305.6019-20) and University of Rijeka scientific grants uniri-tehnic-18-275-1447

custom CNN architecture with the goal of classification. The results and pitfalls of this approach will be discussed and conclusions, along with the possible future work will be presented. The goal of the paper is to test whether the variable-length encoding of SMILES data into matrices, which can serve to conserve space as opposed to fixed-length matrices, is capable of being used for ML problems.

II. SMILES FORMAT DESCRIPTION

Smiles is designed to allow to store the chemical notation of the molecular structure into a format that is easily storable and readable by computer programs. Smiles was developed through US Environmental Protection Agency funding at the Pomona College, Claremont, California, and the Computer Sciences Corporation, Duluth, Minnesota. There are five basic rules used to transform the graphical representation of a chemical bond. These rules pertain to the manner of denoting atoms and bonds (A), simple chains (B), branches (C), rings (D), and charged atoms (E). An example is given below with Figure 1 depicting an ibuprofen molecule. Using the rules that will be described in this section, the molecule in question can be written as CC(C)Cc1ccc(cc1)C(C)C(O)=O.

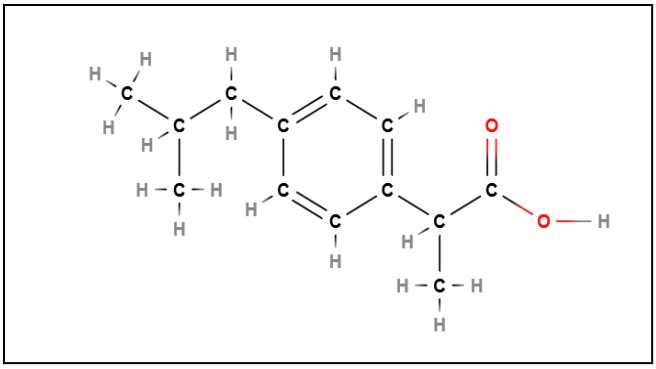

Fig. 1. Example molecule used - ibuprofen

A. Atoms and Bonds

Smiles format supports all the elements present in the periodic table, with each element being denoted using its atomic symbol. If the symbol is written with an upper-case letter, it represents a non-aromatic atom, while a lower-case letter represents an aromatic atom. For those atomic symbols containing more than one letter, the letters beyond the first one are written as lower-case. The atomic bonds are written according to Table I.

TABLE I. BOND DEPICTIONS PER BOND TYPE IN SMILES FORMAT

Bond type	Depiction
Single bond[a]	-
Double bond	=
Triple bond	#
Aromatic bond	*
Disconnected structures	.

[a] *single bonds are default and need not be entered, so the connection between two non-aromatic carbon atoms can be written as CC instead of C-C*

B. Simple Chains

Simple chains are represented by combining atomic symbols and bond symbols. The SMILE structures are hydrogen-suppressed – meaning no hydrogen symbols are used. When software is used to recreate the connections, all bonds unsatisfied with the depicted symbols are assumed to be hydrogen bonds. If the users do specify the hydrogen bonds, they can be interpreted – but it is then assumed that all of the hydrogen bonds have been included in the depiction. As an example, Ethene can be written as C=C, without the hydrogen bonds, or HC(H)=C(H)(H) with the hydrogen bonds. Examples follow in Table II.

TABLE II. BOND DEPICTIONS PER BOND TYPE IN SMILES FORMAT

Molecule	Symbol	SMILES
Ethane	CH_3CH_3	CC
Ethene	CH_2CH_2	C=C
Sodium Chloride	NaCl	Na.Cl
Bromomethane	CH_3Br	CBr

One note to be made is possible interpreter confusion. If we take scandium as an example, the symbol Sc may be interpreted as a Sulfur atom connected to an aromatic carbon with a single boned. To address this, the symbol for scandium (and similar elements) should be written as [Sc].

C. Branches

A branch is depicted by placing the branch symbol from Table I in between parenthesis. The symbol in the parenthesis depicts the further connections, while the bond symbol follows behind the left parenthesis. Some examples are given in Table III.

TABLE III. EXAMPLES OF BRANCHES IN SMILES

Molecule	Example
2-Methylbutane	CC(CC)C
Nitrobenzene	c1c(N(=O)=O)cccc1
2-Propanol	CC(O)C

D. Rings

Ring structures are identified by using numbers to identify the opening and closing ring atom. Chemicals that have multiple uses different numbers for each ring. If the ring is terminated by a double, single, or aromatic bond, the appropriate bond symbol is placed before the ring closure number.

TABLE IV. EXAMPLES OF RINGS IN SMILES

Molecule	Example
Cyclohexene	C=1CCCCC1
Ethyloxirane	C1OC1CC
Naphthalene	c1cc2ccccc2cc1

E. Charged atoms

The valence of the atoms can be automatically assigned by most of the interpreters, but in case that the internal valence knowledge needs to be overridden this can be done by adding the brackets enclosing the charge on the atom following the symbol. The exact charge of the atom in question is given as the number with a sign indicating charge before it – with the "+" sign signifying positive, and the "-" sign signifying negative charge. Examples are given in Table V.

TABLE V. EXAMPLES OF CHARGED ATOM NOTATION IN SMILES

Molecule	Example
Ionized form of propanoic acid	CCC(=O)O{-1}[b]
1-Carboxylmethyl pyridinium	c1ccccn{+1}1CC(=O)O

[b] In the case where the atom has a single charge, the number of charges can be omitted by writing {-} or {+} instead of {-1} or {+1}, respectively

III. SMILES Dataset Transformation

To enable the use of the CNNs on the SMILES data the encoding must be performed to transform them into the matrix-shaped data. This is necessary as CNNs work by performing multi-dimensional convolution with the weight filters, which are adjusted during the backpropagation process, to determine the perform the required tasks of classification/regression/detection or others.

In this paper, to achieve this a one-hot matrix is proposed. One-hot matrix is constructed by taking a set of symbols on one of the axes, with the other ax representing the position of the value. For example, a matrix that would encode a string "AABABDC" would appear as given in Table VI, and mathematical matrix form in Equation 1:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (1)$$

TABLE VI. Example of a One-Hot matrix

		Symbol			
		A	***B***	***C***	***D***
Position	***0***	1	0	0	0
	1	1	0	0	0
	2	0	1	0	0
	3	1	0	0	0
	4	0	1	0	0
	5	0	0	0	1
	6	0	0	1	0

With the format explained, we can start discussing the algorithm that transforms a SMILES string into a one-hot encoded matrix.

First, the number of possible symbols needs to be discussed. As explained previously in the previous section, the possible symbols in SMILES can consist of:

- atomic symbols,
- connections,
- open and closed parenthesis signifying branches,
- number signifying rings, and
- charge symbols.

With the above, we can start determining the maximum possible number of symbols. There are 118 possible chemical symbols, a number that can be doubled to assure that aromatic atoms may be written. Then, there are five possible bond symbols. For the number of rings, we will set the maximum number of rings to 9 (giving us symbols 1 through nine), and we will assume the maximum charge of an atom can be 3. This would give a matrix that is 253 symbols wide. Still, this is unnecessarily large – the probability of all elements being needed for encoding is very low. As the matrix height cannot be predicted, as it depends on the length of the SMILES strings, it needs to be determined dynamically. For this reason, it is best to optimize the matrix size by performing the dynamic allocation of the matrix elements.

The first part of the algorithm is determining the symbol list in the SMILES string. As an example, we will take the following SMILES string:

$$CC(=O)C[Au]c1ccccc1N, \quad (2)$$

representing the molecule given in Figure 2.

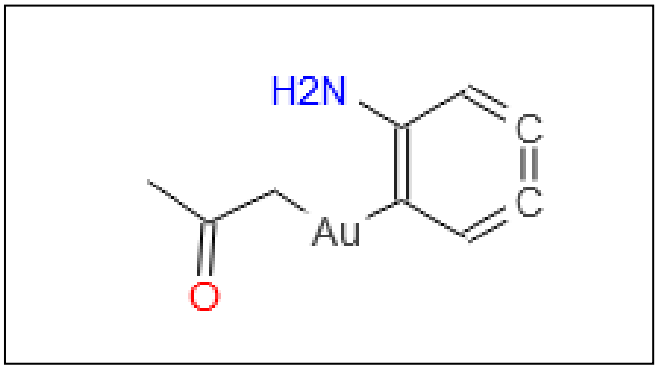

Fig. 2. Graphical representation of the example SMILES molecule CC(=O)C[Au]c1ccccc1N

The algorithm will perform the *separation* operation, according to the following rules:

- if the character is alphanumerical, a bond character (refer to Table I), or a parenthesis, separate is as a unique character, and
- if the symbol read is "{" or "[", read the characters until reaching a "}" or "]" symbol respectively.

The algorithm will iterate over each of the data points and separate the SMILES into an array, per example:

$$CC(=O)C[Au]c1ccccc1N \rightarrow [C, C, (, =, O,), C, Au, c, 1, c, c, c, c, 1, N]. \quad (3)$$

The algorithm will perform two passes – in the first pass, it will determine all the unique symbols in the dataset, as well as the maximum number of symbols in the dataset. While the above is algorithmically inefficient ($O(n^2)$), it allows us to create matrices that all have equivalent dimensions. This is important because some CNN algorithms require that all the input matrices have the same size [24]. For the above example, the list of unique symbols is $[C, (, =, O,), Au, c, 1, N]$, while the maximum length of the symbol array equals 15. By placing the symbol list horizontally, and the positioning vertically we can define the matrix of dimensions 9x15, which is shown in Table VII to visualize the positioning. This matrix essentially tells which symbol is found in which position of the SMILES string, allowing us to contain the spatial relations and ordering of symbols in the matrix form.

In addition to the presented matrix, the algorithm should read any class or regression values that will be the aim of the CNN algorithm. Multiple such values can be stored per run, allowing for the creation of datasets that may be used for multiple classification/regression models. Those values can be stored alongside the one-hot encoded SMILES string inside the so-called NumPy array file (NPY). These files are designed to store tensor data within NumPy arrays and allow the data to be loaded more quickly into ML models when using libraries such as Tensorflow [25]. For longer-term

storage, or in case storage space is critical, NPZ format can be used to compress multiple arrays and save on storage space [26].

TABLE VII. EXAMPLES OF RINGS IN SMILES

		Symbol								
		C	***(***	***=***	***O***	***)***	***Au***	***c***	***1***	***N***
Position	***0***	1	0	0	0	0	0	0	0	0
	1	1	0	0	0	0	0	0	0	0
	2	0	1	0	0	0	0	0	0	0
	3	0	0	1	0	0	0	0	0	0
	4	0	0	0	1	0	0	0	0	0
	5	0	0	0	0	1	0	0	0	0
	6	1	0	0	0	0	0	0	0	0
	7	0	0	0	0	0	1	0	0	0
	8	0	0	0	0	0	0	1	0	0
	9	0	0	0	0	0	0	0	1	0
	10	0	0	0	0	0	0	1	0	0
	11	0	0	0	0	0	0	1	0	0
	12	0	0	0	0	0	0	1	0	0
	13	0	0	0	0	0	0	1	0	0
	14	0	0	0	0	0	0	0	1	0
	15	0	0	0	0	0	0	0	0	1

IV. CNN APPLICATION

The CNN is created to be applied to the Toxicology in the 21st Century (TOX21) dataset [23]. TOX21 dataset contains information on the toxicity of almost 8000 compounds. The dataset consists of 7831 data points, measuring qualitative toxicity measurements on 12 different targets – including stress response pathways and nuclear receptors. This was done with the goal of the binary classification of the NR-AR parameter, where NR stands for neuroreceptor, and AR standing for the androgen receptor. In other words, it's a value that defines whether there is a neuro or androgen receptor response to a given compound, encoded as "*1*" (response recorded) or "*0*" (no response recorded) This parameter was chosen because all datapoints contained it, and it was equally distributed amongst classes. No further thought was given to the selection of the classification goal, as the goal of the presented research is to test the possibility of using the described methodology, and not to achieve state-of-the-art results on the TOX21 dataset.

After performing the dataset transformation into matrix format, according to section III. we obtain the dimensions of the input matrix to be 95x248 (number of unique symbols x maximum symbol array size). This allows us to design a CNN architecture. Starting with the input size of $95x248$, we can apply the formula to calculate the next layer size, according to parameters: the spatial extent of the filter kernel F, stride S, number of filter kernels K, and padding P, using equation [27]:

$$W' = \frac{W - F + 2P}{S} + 1. \quad (4)$$

Where W' is the dimension of the matrix resulting from convolution (in either direction), and W is the current dimension in the same direction. In this paper, the same values of the convolutional parameters are used in the entire network, and given as: the number of filters set to 64, spatial extent set to 16, padding set to 0, and stride set to 1, in the first layer the input transforms as:

$$(95,248,1) \rightarrow (90,243,10). \quad (5)$$

Repeating the process gives as a CNN that is 7 convolutional layers deep, with the final matrix size of $(5,158,64)$. A schematic display of the network is given in Figure 3.

This final layer is densely connected with a single neuron, containing a sigmoid activation function. CNN has other hyperparameters which define the architecture that needs to be set. The loss function that evaluates the quality of the network model while training is binary cross-entropy, the solver used is adam [28]. Two varied hyperparameters are the number of epochs and the number of batch sizes. The number of epochs tested is 5, 10, 20, 50, and 100, while the batch sizes are set to 1, 8, 16, and 64. While a more complex network, with a wider hyperparameter search, has the potential for achieving better results, a relatively simple architecture was chosen to allow for a faster training process, as the main focus of the research is to be proof-of-concept of utilizing CNNs with the described dataset transformation.

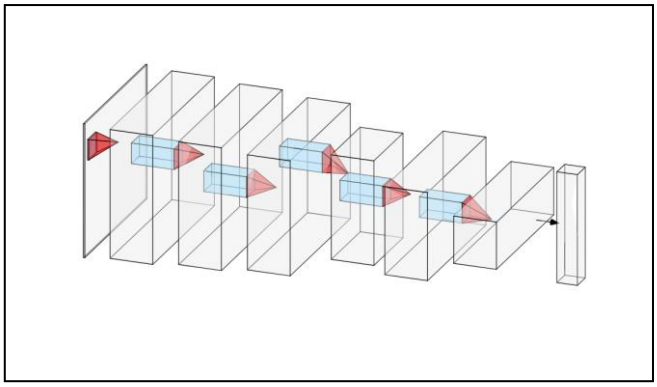

Fig. 3. Schematic display of the used CNN.

The classification quality of the network is assessed using the AUC metric which evaluates the ability of the classifier to distinguish between two classes, with the higher value indicating a better classification [29]. The dataset is split into 80:10:10 train/test/validation sets, with the distribution being 6265 data points for training. 783 data points for testing and 783 data points for validation.

V. SOFTWARE USED

Implementation is performed in Python 3.7.10, distributed within the Anaconda stack. Libraries used for data conversion are Pandas (version 1.3.2) for original data load and NumPy (version 1.20.3) for data conversion and saving. The described CNN is implemented using Tensorflow library (2.3.0), with Keras Deep Learning API (version 2.4.0), data again being loaded through the aforementioned NumPy library.

VI. RESULTS AND DISCUSSION

This section will present the results of the research. First, the results achieved via CNN, for each hyperparameter combination will be shown and discussed, followed by the discussion on the execution times of the SMILES dataset transformation algorithm.

A. Classification Results

The achieved results are given in Table VIII, with the achieved AUC score for the presented CNN at each epoch and batch size that was tested during the research. Observing the scores obtained, we can see that they reach maximum scores for 50 epochs of training. They show a drop for 100 epochs of training, indicating overfitting which suggests that 50 epochs are a near-optimal number of training epochs. Observing scores per batch size, we can see that lower batch sizes provide better results. This may be because smaller batch sizes can exhibit a regularization effect [30]. Regularization can lower the influence of the highly affluent inputs, allowing for models with better generalization.

TABLE VIII. RESULTS CNN MODEL ACHIEVED ON THE TRANSFORMED TOX21 DATASET

Hyperparameters		**AUC**
Epochs	***Batch Size***	
10	1	0.75
	8	0.73
	16	0.72
	64	0.71
20	1	0.84
	8	0.84
	16	0.82
	64	0.79
50	1	0.91
	8	0.92
	16	0.88
	64	0.88
100	1	0.90
	8	0.89
	16	0.87
	64	0.88

B. Algorithm executions speed

The mean execution time of the conversion algorithm is $\bar{T} = 10.99\ min$, with a standard deviation of $\sigma = 0.1$, for $N = 10$ executions on an i7-1065G7 CPU clocked at 1.30GHz with turbo boost disabled, coupled with 16GB of available RAM. It should be noted that the algorithm in the current state is single-threaded, so significant improvements may be made with multi-threaded execution. Given the presented numbers, by dividing the total execution time with the number of samples in the TOX21 dataset, it can be concluded that the execution time is approximately 0.08 seconds per dataset point, which can be used to assume the times for other databases.

VII. CONCLUSIONS

In this paper, the authors demonstrated a methodology for the creation of matrix-shaped datasets from the datasets in which the chemical compound molecule shapes are stored using SMILES format. Using the transformation on the TOX21 dataset, the authors could apply a CNN architecture with the classification goal, achieving a classification score of 0.92 for the problem of classifying compounds that influence the neural and androgen receptors. Authors conclude that the algorithm performance is satisfactory, with the average time to process a single datapoint, transform it and store it is below 0.1 second.

From the presented results it can be concluded that a successful models can be developed using CNNs when the described methodology is applied to the SMILES dataset transformation into a matrix shape. While the classification results are relatively low, it is possible they could be improved with a more complex CNN architecture, such as VGG19 or Xception. The timing of the algorithm is considered satisfactory by the authors, but with definite room for improvement with the introduction of multi-threaded execution and other code optimizations.

The described are the elements of future work relating to research in this paper, along with the application of the described methodology on novel datasets that are stored and distributed using SMILES format notation.

REFERENCES

[1] I. Lorencin, N. Anđelić, J. Španjol, and Z. Car, "Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis," *Artificial Intelligence in Medicine*, vol. 102, p. 101746, 2020

[2] A. S. Panayides, A. Amini, N. D. Filipovic, A. Sharma, S. A. Tsaftaris, A. Young, D. Foran, N. Do, S. Golemati, T. Kurc, K. Huang, K. S. Nikita, B. P. Veasey, M. Zervakis, J. H. Saltz, and C. S. Pattichis, "AI in Medical Imaging Informatics: Current Challenges and Future Directions," IEEE Journal of Biomedical and Health Informatics, vol. 24, no. 7, pp. 1837–1857, 2020.

[3] C. Hong, I. Jeong, L. F. Vecchietti, D. Har, and J.-H. Kim, "AI World Cup: Robot Soccer-Based Competitions," IEEE Transactions on Games, pp. 1–1, 2021.

[4] S. Baressi Šegota, N. Anđelić, V. Mrzljak, I. Lorencin, I. Kuric, and Z. Car, "Utilization of multilayer perceptron for determining the inverse kinematics of an industrial robotic manipulator," *International Journal of Advanced Robotic Systems*, vol. 18, no. 4, pp. 1–11, Aug. 2021.

[5] N. Anđelić, S. Baressi Šegota, I. Lorencin, V. Mrzljak, and Z. Car, "Estimation of COVID-19 epidemic curves using genetic programming algorithm," Health Informatics Journal, vol. 27, no. 1, p. 146045822097672, 2021.

[6] Z. Car, S. Baressi Šegota, N. Anđelić, I. Lorencin, and V. Mrzljak, "Modeling the Spread of COVID-19 Infection Using a Multilayer Perceptron," Computational and Mathematical Methods in Medicine, vol. 2020, pp. 1–10, 2020.

[7] D. Štifanić, J. Musulin, A. Miočević, S. Baressi Šegota, R. Šubić, and Z. Car, "Impact of COVID-19 on Forecasting Stock Prices: An Integration of Stationary Wavelet Transform and Bidirectional Long Short-Term Memory," Complexity, vol. 2020, pp. 1–12, 2020.

[8] S. Zhao, F. Blaabjerg, and H. Wang, "An overview of artificial intelligence applications for power electronics," 2020.

[9] S. Ge and Y. Zhang, "Application Research of Biochemistry in Life Science Based on Artificial Intelligence," *Advances in Intelligent Systems and Computing Big Data Analytics for Cyber-Physical System in Smart City*, pp. 272–278, 2020.

[10] L. Heo, G. Janson, and M. Feig, "Physics-based protein structure refinement in the era of artificial intelligence," *Proteins: Structure, Function, and Bioinformatics*, 2021.

[11] B. I. Basok, "Digitalization and artificial intelligence in laboratory medicine," *International Journal of Medical Biochemistry*, 2020.

[12] J. Peña-Guerrero, P. A. Nguewa, and A. T. García-Sosa, "Machine learning, artificial intelligence, and data science breaking into drug design and neglected diseases," *WIREs Computational Molecular Science*, vol. 11, no. 5, 2021.

[13] Q. Bai, S. Tan, T. Xu, H. Liu, J. Huang, and X. Yao, "MolAICal: a soft tool for 3D drug design of protein targets by artificial intelligence and classical algorithm," *Briefings in Bioinformatics*, vol. 22, no. 3, 2020.

[14] J. Jiménez-Luna, F. Grisoni, and G. Schneider, "Drug discovery with explainable artificial intelligence," *Nature Machine Intelligence*, vol. 2, no. 10, pp. 573–584, 2020.

[15] B. Mirzai, D. Lim, and G. Moschytz, "Robust CNN templates: theory and simulations," 1996 Fourth IEEE International Workshop on Cellular Neural Networks and their Applications Proceedings (CNNA-96).

[16] C. M. Holden and M. F. Greaney, "Frontispiece: Modern Aspects of the Smiles Rearrangement," *Chemistry - A European Journal*, vol. 23, no. 38, 2017.

[17] Pinheiro, G., Mucelini, J., Soares, M., Prati, R., Da Silva, J. and Quiles, M., 2021. *Machine Learning Prediction of Nine Molecular Properties Based on the SMILES Representation of the QM9 Quantum-Chemistry Dataset*. [online] ACS Publications.

[18] G. A. Pinheiro, J. L. F. D. Silva, M. D. Soares, and M. G. Quiles, "A Graph-Based Clustering Analysis of the QM9 Dataset via SMILES Descriptors," Computational Science and Its Applications – ICCSA 2020 Lecture Notes in Computer Science, pp. 421–433, 2020.

[19] R. V. Deursen, P. Ertl, I. Tetko, and G. Godin, "GEN: Highly Efficient SMILES Explorer Using Autodidactic Generative Examination Networks," 2019.

[20] X. Yang, J. Zhang, K. Yoshizoe, K. Terayama, and K. Tsuda, "ChemTS: an efficient python library for de novo molecular generation," *Science and Technology of Advanced Materials*, vol. 18, no. 1, pp. 972–976, 2017.

[21] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. Cambridge (EE. UU.): MIT Press, 2016.

[22] X. He and Y. Chen, "Transferring CNN Ensemble for Hyperspectral Image Classification," IEEE Geoscience and Remote Sensing Letters, vol. 18, no. 5, pp. 876–880, 2021.

[23] R. R. Tice, C. P. Austin, R. J. Kavlock, and J. R. Bucher, "Improving the Human Hazard Characterization of Chemicals: A Tox21 Update," Environmental Health Perspectives, vol. 121, no. 7, pp. 756–765, 2013.

[24] S.-H. Yoon and H.-J. Yu, "Multiple Points Input For Convolutional Neural Networks in Replay Attack Detection," ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020.

[25] "NumPy," Learning Scientific Programming with Python, pp. 196–293, 2020.NPZ

[26] B. Boashash and V. Sucic, "High Performance Time-Frequency Distributions for Practical Applications," Wavelets and Signal Processing, pp. 135–175, 2003.

[27] T D. Grattarola and C. Alippi, "Graph Neural Networks in TensorFlow and Keras with Spektral [Application Notes]," IEEE Computational Intelligence Magazine, vol. 16, no. 1, pp. 99–106, 2021.

[28] I. Lorencin, S. Baressi Šegota, N. Anđelić, V. Mrzljak, T. Ćabov, J. Španjol, and Z. Car, "On Urinary Bladder Cancer Diagnosis: Utilization of Deep Convolutional Generative Adversarial Networks for Data Augmentation," Biology, vol. 10, no. 3, p. 175, 2021.

[29] H. Yong, J. Huang, D. Meng, X. Hua, and L. Zhang, "Momentum Batch Normalization for Deep Learning with Small Batch Size," Computer Vision – ECCV 2020 Lecture Notes in Computer Science, pp. 224–240, 2020.