# Solubility classification of molecules

*CS-433: Machine Learning Course Project 2*
Malte Franke, Laura Mismetti and Giacomo Mossinelli
*Laboratory of Artificial Chemical Intelligence (LIAC), EPFL, Switzerland*

***Abstract*—In this study, the performance of various machine learning and deep learning models is tested against their ability to classify molecules depending on their solubility. Different molecule representations are used together with classical techniques of feature engineering and hyperparameter optimization. Overall, it is found that more complex models, (GNNs, ChemBERTa, SchNet) have worse performance than XGBoost, while they outperform simpler models (RF, SVMs, ANNs). However, the representations are not expressive enough for describing solubility, leading to an inseparable problem.**

## I. Introduction

The knowledge of chemical properties of small molecules is fundamental to understanding their behavior in different environments, such as the human organism. The accurate prediction of such properties, without prior experimental characterization, would allow a dramatic speed-up in the research of new chemical probes and drugs, that could prevent millions of deaths every year [1].

As part of the ML4science project, this work aims at contributing to the process of predicting the solubility of small molecules. Solubility is the ability of molecules to dissociate into a solution or solvent, and it is an essential property in drug discovery. More specifically, the goal of this work is the classification of molecules based on their Simplified Molecular Input Line Entry Specification (SMILES) string representation into low (0), medium (1), and high (2) aqueous solubility. The 3 classes are defined through the measured Nephelometry values.

## II. Data

Train and test datasets are taken from the Kaggle challenge "1st EUOS/SLAS Joint Challenge: Compound Solubility" [1]. The first contains about 70.000 molecules as SMILES strings with the respective target classes, while the second one contains 30.000 unlabeled SMILES.

One key problem of the challenge is the imbalanced dataset. In total, the dataset contains only about 2.000 molecules of class 0, 2.500 of class 1 while class 2 is made up of around 65.000 molecules. Indeed, more data are available for highly soluble molecules since drug discovery is more interested in their screening. Overcoming the imbalance is necessary to avoid the model classifying molecules as class 2 only. Approaches to handling this issue are presented in Section II-C.

### A. *Molecule Representations*

To classify molecules with a Machine Learning (ML) model, a machine-readable representation is needed. Figure 1 provides an overview of the representations used in this work.

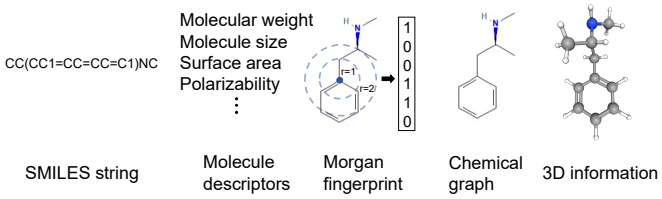


Figure 1. Molecule representations used in this work.

*1) **SMILES**:* The most common way to represent molecules in computational chemistry is through SMILES [2]. They are strings of characters which follow strict rules and precise syntax. Atoms are letters, while special characters enable the representation of more complicated chemistry, e.g. "=" encodes double bonds and ":" aromatic bonds. SMILES can be transformed into other different machine-readable representations.

*2) **Morgan Fingerprint**:* As a baseline, several models have been trained on Morgan fingerprints of molecules, also known as Extended-Connectivity fingerprints (ECFPs) [3], [4]. These circular fingerprints are binary vector representations of molecules that are determined by embedding local information through a hashing function. RDKit has been used to calculate molecular fingerprints with size 2048 based on a neighborhood range of 3 [5].

*3) **Molecular Descriptors**:* Another option is to use features such as molecular size, weight, surface area or quantum mechanical descriptors as input for the classifier. Mordred was used to create such representation [6], resulting in about 1800 features per molecule, both continuous and categorical. In a pre-processing step, columns with NaN values are removed and, to have a more homogeneous dataset, the categorical features are deleted too. About 600 features are left to train ML models. Afterwards, all continuous variables are standardized. Since a lot of features are skewed, a logarithmic transformation is imposed on about 170 columns. A column can be considered skewed if it contains only non-negative values and if $|Sk| = \left| \frac{\sum_{i=1}^{n} (x_i - \bar{x})^3}{n\sigma^3} \right| > 1$ where $\sigma$ is the standard deviation and $\bar{x}$ the mean.

*4) **3D Information**:* Molecules can be described via their atom types and spatial 3D information. In physical chemistry, this information is used (often in combination with charge and spin) as input to the Schrödinger equation. Solving this equation (or approximations thereof) allows for the calculation of molecule energies and, subsequently, the properties of molecules. The 3D structures were obtained by RDKit estimates [5].

*5) **Graphs**:* Molecules can naturally be represented as graphs where atoms correspond to nodes and bonds to edges. Each node and edge has an associated feature vector. In a chemistry context, nodes are often featurized by their atom type, number of bonds, whether it is part of a ring, and by the number of bonding Hydrogen atoms. On the other hand, edges can contain information about the type of bond, if it is part of a ring, and stereochemical information. However, the graph features can be enriched with relevant information and are not limited by the above-mentioned featurization.

### B. Exploratory Data Analysis

Once the given molecules are transformed into an adequate representation, it is useful to perform exploratory, qualitative analysis of the obtained dataset.

*1) **UMAP**:* In order to find clusters or relationships between classes, the dimensionality reduction tool UMAP [7] (similar to t-SNE [8]) is used. In this specific case, it is possible to infer that the dataset can be randomly split since train and test data are overlapping (see Fig.2). Moreover, it is not easy to distinguish the classes in an appropriate way, since they all lay in similar neighborhoods. No particular clustering can be observed (see Fig. 2 and 3).

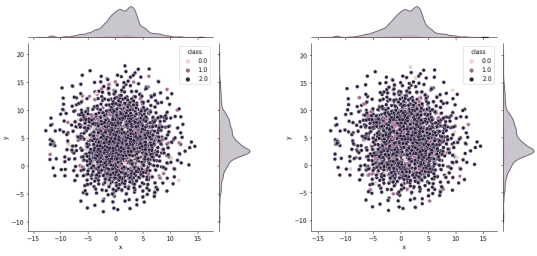

Figure 2. UMAP comparison of Morgan fingerprints with example train (left) and local test set (right). It is possible to notice that the two figures are practically indistinguishable.

*2) **Scaffold analysis**:* The Bemis-Murcko scaffold is defined as the core structure of a molecule, obtained by removing side chains or R-groups [9]. The comparison of the scaffolds in train and test set allows for assessing if the types of molecules in these sets are similar. In this work, the similarity between two molecules is defined by their Tanimoto distance [10]. Two SMILES are declared equal if their similarity is above $95\%$. The train set contains 25456 unique scaffolds, while the test set is made up of 8626 different scaffolds. However, only 3600 of the latter are contained in the train set too. A model makes good predictions only if it generalizes to new types of molecules. This preliminary analysis shows that the test set contains many unknown scaffolds, which makes prediction a challenging task.

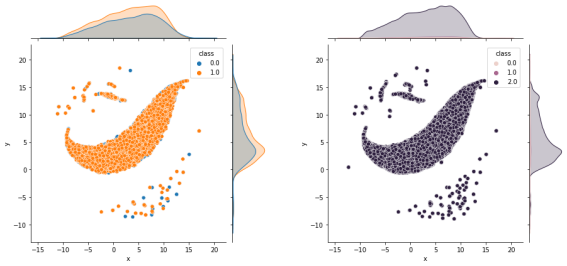

Figure 3. UMAP representation of molecular descriptors of the train set. It is possible to observe that the blue dots (class 2 elements) completely cover the others. All the classes have similar distributions.

### C. Handling Data Imbalance

The datasets present a strong imbalance of the three classes and this problem needs to be carefully addressed. Three approaches were followed to handle this issue.

*1) **Under/Over-sampling**:* The most common way to face imbalanced classes is under- or over-sampling. A defined percentage of samples of the majority class is randomly removed, while a certain amount of samples from the minority classes is duplicated. Usually, these operations are performed to reach a comparable percentage of data for the three classes in the dataset, $33\% - 33\% - 34\%$ are used in this work. However, discarding data is a debatable choice. In addition, the introduction of copies doesn't always imply an improvement of predictions and requires careful data handling during training.

*2) **SMILES augmentation**:* When SMILES are used as direct input of an ML model, another way to handle class imbalance is possible. Since atoms in SMILES strings are numbered, changing their order in the enumeration produces a copy of the molecule with a different SMILES representation, a so-called *rotated SMILES* [11]. Through this method, unique rotations of training molecules can be created to augment the dataset. The number of possible augmentations depends on the number of heavy atoms in the molecule. Unfortunately, the usage is limited to string-based ML models as rotated SMILES result in the same fingerprints and descriptors. In the present study, all possible molecule rotations have been added for class 0 and class 1, while only 5 have been used for class 2. The latter is necessary to learn a model that is invariant with respect to the applied transformation for all the classes.

*3) **Weighted classes**:* Another method is introducing weights for the loss function. Every sample is weighted depending on the class it belongs to. Specifically, the weights are calculated as $w_i = 1 - \frac{n_i}{N}$ where $N$ is the total amount of elements and $n_i$ the number of elements belonging to class $i = 0, 1, 2$.

## III. Classical ML Models

Building on the above-mentioned representations, the following ML models are trained to predict the solubility class of molecules. Ensembling with $n = 5$ is used to increase the predictive power and robustness of the models [12].

### A. Random Forest

The Random Forest (RF) classifier is an ensemble of decision trees acting on feature entries, relating them to the property at hand [13]. In this work, the Scikit-learn implementation of RF is used with parameters chosen via randomized cross validation [14].

### B. Support Vector Machine

Support vector machines (SVMs) are models that try to find the hyperplane that maximally separates the classes [15]. The SVM implementation of Scikit-learn with a Gaussian RBF kernel is used, which allows for fitting highly non-linear functions by using the kernel trick to project the inputs to a so-called hyperspace [14]. This process can be seen as a very efficient feature extension. The used parameters are chosen with a randomized cross validation.

### C. Artificial Neural Networks

Being powerful and flexible ML models, artificial neural networks (ANNs) are popular for many data science applications. In this work, an ANN with one hidden layer of 300 nodes, dropout layers with $p = 0.5$ and ReLU activation functions is implemented in the deep learning framework PyTorch [16], [12], [17]. Adam is used to optimize the weights of the networks [18]. Early stopping with a patience of 30 epochs is adopted to obtain the best model with respect to the test loss [12].

### D. XGBoost

XGBoost uses gradient boosting, an ML technique that combines multiple weak models, such as decision trees, to create a stronger one [19]. More specifically, it trains the weak models in a sequential manner, with each of them attempting to correct the mistakes made by the previous ones. The final model is a weighted sum of the weak models, where the weights are determined by the optimization of an objective function. In this work both AUC (that performs a one vs. rest classification) and multiclass logloss are used as evaluation metrics for validation data, respectively with softprob and softmax objectives. Optimal parameters of the model are chosen by 5-fold cross-validation.

## IV. Deep Learning Models

### A. SchNet

SchNet is a model that uses 3D information of molecules to learn properties and was mainly developed for the prediction of quantum mechanical properties [20]. RDKit estimates of the 3D structure of the molecules are calculated on the basis of the SMILES to train this model [5].

### B. ChemBERTa

ChemBERTa is used to classify molecules [21], [22]. It is based on a Transformer architecture developed in the domain of Natural Language Processing (NLP) [23]. It acts directly on the text-based SMILES representation to make predictions about the molecule. Models that are pre-trained on large datasets (supervised or unsupervised) learn the underlying "language" of chemistry and can be fine-tuned on a downstream task. The pre-trained "DeepChem/ChemBERTa-10M-MTR" model is used (https://huggingface.co/DeepChem/ChemBERTa-10M-MTR).
Training based on SMILES string makes data augmentation possible (see Section II-C2).

### C. Graph Neural Network

Graph neural networks (GNNs) have shown great success in molecule property prediction in recent years [24]. GNNs directly act on graph structure of molecules, propagating node and edge-level information through a procedure known as message-passing [24]. A representation useful for the underlying task is learned end-to-end. The GNN implementation provided in the Chemprop framework released by [25] is used.

## V. Results

Table I shows the obtained results measured on the quadratic Cohen's kappa $\kappa$, an important metric for inter-rater reliability in classification tasks bounded by -1 (complete disagreement) and 1 (correct prediction). Values of 0 resemble random choice.

First of all, Morgan fingerprints are used to represent molecules in combination with classical ML models, in particular RF, SVM, ANN, and XGBoost. The class imbalance has been handled by using loss weights (see II-C3). However, the $\kappa$ value is low in all those cases, so that a different representation of the molecules is adopted. The same models are tested by using molecular descriptors (II-A3) as input, but no relevant improvement can be seen with only the exception of XGBoost, which reaches $\kappa = 0.09689$.
Consequently, since simple models are not able to correctly classify molecules, more complex ones are studied. SchNet is used with 3D representations of molecules and downsampling of class 2 II-C1, which results in a slightly worse accuracy. On the other hand, ChemBERTa, trained by using weights to handle the class imbalance,

Table I

| MODELS | Quadratic Cohen's kappa $\kappa$ |
|---|---|
| Morgan FP + RF | 0.00032 |
| Morgan FP + SVM | 0.00996 |
| Morgan FP + ANN | 0.03928 |
| Morgan FP + XGBoost | 0.01889 |
| Mol. descriptors + RF | 0.00572 |
| Mol. descriptors + SVM | 0.01283 |
| Mol. descriptors + XGBoost | 0.09689 |
| 3D + SchNet | **0.06267** |
| SMILES + ChemBERTa | **0.09014** |
| augmented SMILES + ChemBERTa | 0.00000 |
| GNN | 0.0124 |
| Mol. descriptors + PCA + transforms* + XGBoost | 0.09011 |
| Mol. descriptors + FP + transforms* + XGBoost | **0.11211** |
| Ensemble model | 0.10929 |

Kaggle score of the most important models.
($*$ transformations described in Section II-A3.)

presents performance comparable to XGBoost. An additional study is performed to try to manage class imbalance in a different way: SMILES augmentations are introduced in the dataset used for ChemBERTa. However, this leads to the worst kappa value. The "DeepChem/ChemBERTa-77M-MTR" model that is pre-trained over more data is also used but gives worse results (https://huggingface.co/DeepChem/ChemBERTa-77M-MTR). The problem in both cases can come from the pre-training of Chemberta, which is done only on canonicalized SMILES without any augmentation, so that rotations are only adding noise in the dataset. Finally, the GNN model doesn't improve the solubility predictions as well. The reason behind these low results could be that the used molecules representations cannot capture differences that are significant to determine solubility, as can be seen in Figure 3. It is clear that the tested deep learning models (SchNet, ChemBERTa and GNNs) lead to worse or comparable results to the previously described XGBoost model, but they are better classifiers than simple models.

Since the combination of molecular descriptors and XGBoost outperforms every other model, the focus now goes on its improvement. Feature engineering is applied and the hyperparameters are studied. In particular, the inclusion of both Morgan fingerprints and molecular descriptors and the application of all the transformations described in II-A3 bring to $\kappa = 0.11211$ by using softmax as objective and logloss as evaluation metric. Moreover, using PCA on molecular descriptors, it is possible to reduce the number of features to 100 with $97.5\%$ of variance explained (Figure 4). This leads to $\kappa = 0.04358$. The application of the transformations described in II-A3 allows to reach $\kappa = 0.09011$.

In the end, as highlighted in Table I, the three models with best performances are using different molecule representations (Molecular descriptors, 3D structure, SMILES), meaning they are predicting solubility based on different information about the molecule. Hence, in order to obtain a more robust result, an attempt in combining

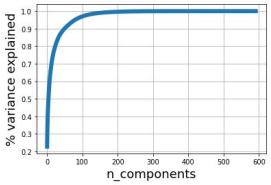


Figure 4. Explained variance of PCA on molecular descriptors (columns with NaN values and categorical columns not considered) per number of features.

the predictions of XGBoost, SchNet and ChemBERTa is performed, as described in [26], so that an ensemble model is created. In particular, the final prediction is done by choosing one of the minority classes (i.e. 0 and 1), if one of the models predicts it, and a majority vote in the other cases. The result obtained is $\kappa = 0.10929$.

## VI. Conclusion

In this work, several ML methods are used to classify molecules based on their predicted solubility. However, none of the tested approaches yields satisfactory results. It is evident that the classical ML models used are not able to fulfill the proposed classification task, while deep learning models show slightly better results. The best performance is obtained with XGBoost, when using both Morgan fingerprints and molecular descriptors, together with transformations described in II-A3. However, it reaches only $\kappa = 0.11211$. The use of PCA simplifies the model by eliminating possible collinearities and substantially reduces the training time without dramatically impacting the performance. More specifically, the time required to train this model is approximately 20% of XGBoost and is also far lower than the time needed to train the complex models, which need GPUs during training.

Using an ensemble model can sometimes lead to performance improvement. Combining different representations of molecules can be useful, as they focus on different aspects that together can be more expressive for certain properties. However, the created ensemble model only reach a similar accuracy to the best XGBoost model.

The applied techniques to handle the class imbalance can improve the predictions, except for SMILES augmentation used with ChemBERTa, which dramatically reduces the performance of the model.

In general, the reason behind the low performances of the proposed models may be that the used molecules representations cannot capture significant differences to determine solubility, as can be seen in Figure 3. Indeed, subtle changes in molecules structure, like the addition of a specific functional group, can drastically change their solubility. Further works should focus on finding other molecules representations that are able to make the classes distinguishable by a classifier.

## REFERENCES


[1] J. M. R. H. W. Andrea Zaliani, Jing Tang, "1st euos/slas joint challenge: Compound solubility," 2022. [Online]. Available: https://kaggle.com/competitions/euos-slas

[2] D. Weininger, "Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules," *J. Chem. Inf. Comput. Sci.*, vol. 28, pp. 31–36, 1988.

[3] H. L. Morgan, "The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service." *Journal of Chemical Documentation*, vol. 5, pp. 107–113, 1965.

[4] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of chemical information and modeling*, vol. 50, pp. 742–54, 05 2010.

[5] RDKit, "Open-source cheminformatics," 2022. [Online]. Available: http://www.rdkit.org

[6] H. Moriwaki, Y. Tian, N. Kawashita, and T. Takagi, "Mordred: A molecular descriptor calculator," *Journal of Cheminformatics*, vol. 10, 02 2018.

[7] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2018. [Online]. Available: https://arxiv.org/abs/1802.03426

[8] L. van der Maaten and G. Hinton, "Viualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 11 2008.

[9] G. W. Bemis and M. A. Murcko, "The properties of known drugs. 1. molecular frameworks." *Journal of medicinal chemistry*, vol. 39 15, pp. 2887–93, 1996.

[10] J. Godden, L. Xue, and J. Bajorath, "Combinatorial preferences affect molecular similarity/diversity calculations using binary fingerprints and tanimoto coefficients," *Journal of chemical information and computer sciences*, vol. 40, pp. 163–6, 02 2000.

[11] J. S. P. O. e. a. Arús-Pous, J., "Randomized smiles strings improve the quality of molecular generative models," *J Cheminform*, vol. 11, no. 71, 2019.

[12] I. Goodfellow, Y. Bengio, and A. Courville, *"Deep Learning"*. MIT Press, 2016, http://www.deeplearningbook.org.

[13] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 10 2001.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[15] B. Schölkopf and A. Smola, *"Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond"*, ser. Adaptive Computation and Machine Learning. Cambridge, MA, USA: MIT Press, 2002.

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, ""dropout: A simple way to prevent neural networks from overfitting"," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: http://arxiv.org/pdf/1412.6980v9

[19] T. Chen and G. C., "Xgboost: A scalable tree boosting system," 2016. [Online]. Available: https://arxiv.org/pdf/1603.02754.pdf

[20] K. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions," 06 2017.

[21] S. Chithrananda, G. Grand, and B. Ramsundar, "Chemberta: Large-scale self-supervised pretraining for molecular property prediction," 2020. [Online]. Available: https://arxiv.org/abs/2010.09885

[22] W. Ahmad, E. Simon, S. Chithrananda, G. Grand, and B. Ramsundar, "Chemberta-2: Towards chemical foundation models," 2022. [Online]. Available: https://arxiv.org/abs/2209.01712

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 06 2017.

[24] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," 2017. [Online]. Available: https://arxiv.org/abs/1704.01212

[25] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen, and R. Barzilay, "Analyzing learned molecular representations for property prediction," *Journal of Chemical Information and Modeling*, vol. 59, 07 2019.

[26] T. K. K. M. e. a. Chen, CH., "Comparison and improvement of the predictability and interpretability with ensemble learning models in qspr applications," *J Cheminform*, vol. 12, no. 19, 2020.