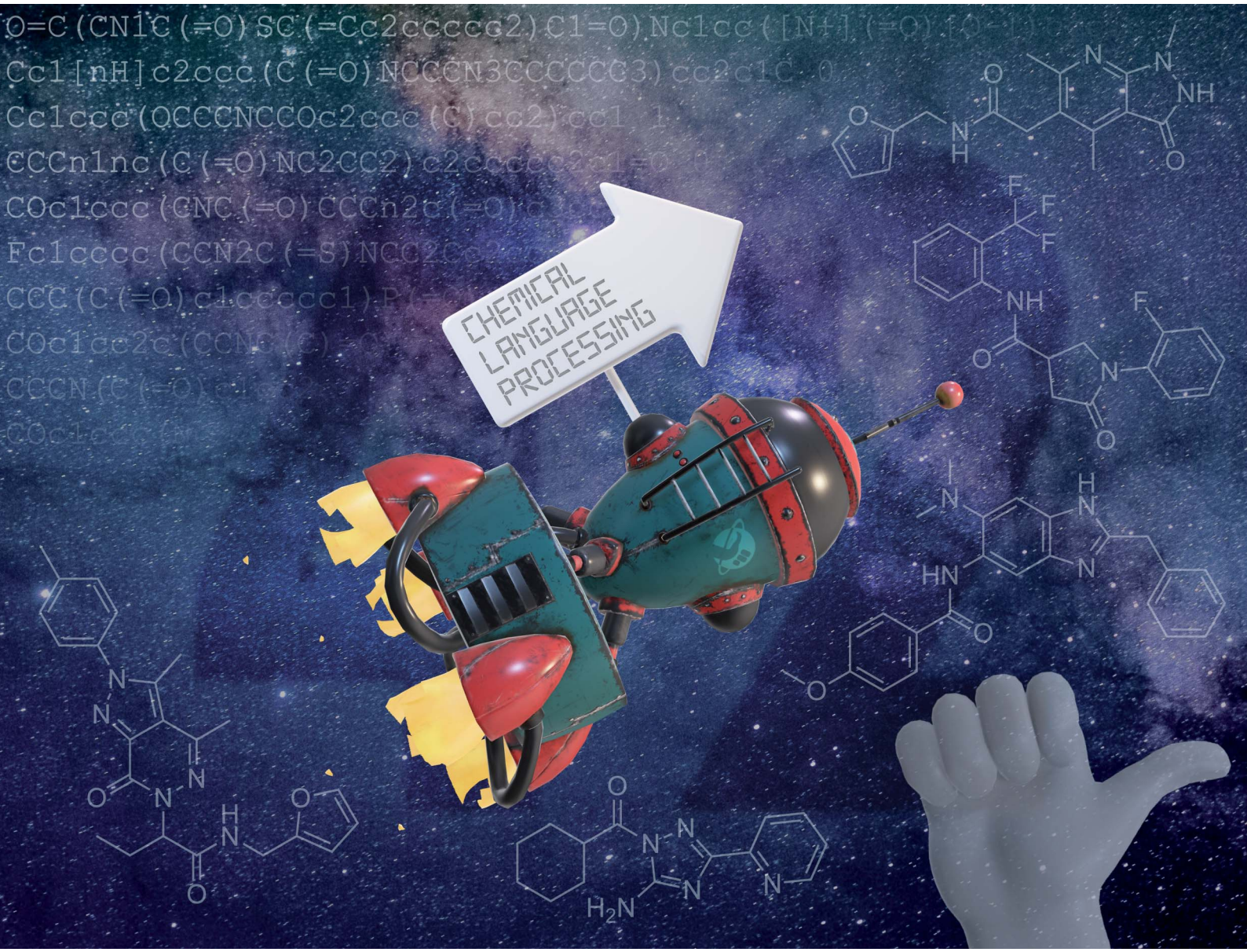
Volume 4
Number 2
February 2025
Pages 291-574

Digital Discovery

rsc.li/digitaldiscovery



ISSN 2635-098X

ROYAL SOCIETY OF CHEMISTRY

COMMUNICATION

Rıza Özçelik and Francesca Grisoni

A hitchhiker's guide to deep chemical language processing for bioactivity prediction

Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

COMMUNICATION

View Article Online
View Journal | View Issue

Cite this: *Digital Discovery*, 2025, **4**, 316

A hitchhiker's guide to deep chemical language processing for bioactivity prediction†

Rıza Özçelik [ab] and Francesca Grisoni [*ab]

Received 26th September 2024
Accepted 13th December 2024

DOI: 10.1039/d4dd00311j

rsc.li/digitaldiscovery

Deep learning has significantly accelerated drug discovery, with 'chemical language' processing (CLP) emerging as a prominent approach. CLP approaches learn from molecular string representations (*e.g.*, Simplified Molecular Input Line Entry Systems [SMILES] and Self-Referencing Embedded Strings [SELFIES]) with methods akin to natural language processing. Despite their growing importance, training predictive CLP models is far from trivial, as it involves many 'bells and whistles'. Here, we analyze the key elements of CLP and provide guidelines for newcomers and experts. Our study spans three neural network architectures, two string representations, three embedding strategies, across ten bioactivity datasets, for both classification and regression purposes. This 'hitchhiker's guide' not only underscores the importance of certain methodological decisions, but it also equips researchers with practical recommendations on ideal choices, *e.g.*, in terms of neural network architectures, molecular representations, and hyperparameter optimization.

1 Introduction

Machine learning has accelerated drug discovery.[1,2] The prediction of biological properties, such as the interaction with macromolecular targets, has been pivotal in this context, *e.g.*, for hit finding and lead optimization.[2–5] While several 'flavours' of machine learning exist (*e.g.*, graph neural networks),[6,7] deep learning models that use string representations of molecules, like Simplified Molecular Input Line Entry System (SMILES)[8] and Self-Referencing Embedded Strings (SELFIES),[9] have drawn particular interest.[10–12] Such deep 'chemical language' processing approaches apply methods akin to natural language processing to learn from molecular string representations.[13,14]

Molecular string representations (*e.g.*, SMILES[8] and SELFIES,[9] among others[15–18]) have found widespread application in cheminformatics and related fields.[13,19] They convert two-dimensional molecular information into strings, by traversing the molecular graph and annotating atom and bond information with dedicated symbols (Fig. 1a). Deep 'chemical language processing' (CLP) models are then trained to map the chemical information in such strings to a property to be predicted, *e.g.*, a ligand interaction with a target or toxicological properties. Once trained, CLP models can be applied prospectively, for instance, to screen large molecular libraries in search of molecules with desirable properties.[20,21]

Developing predictive CLP models is far from trivial[22,23] and it requires many choices to be made,[24] *e.g.*, in terms of molecular string representations and their encoding, and of neural network architectures and their hyperparameters. Each such choice might affect the model performance. Stemming from these observations, this 'hitchhiker's guide' aims to discover best practices in the field, and provide a guideline for what choices to make when training CLP models for bioactivity prediction. Here, we derive our insights from a systematic analysis of three deep learning architectures, two molecular string representations, and three encoding approaches on ten datasets spanning regression and classification tasks.

Ultimately, this 'hitchhiker's guide' provides some 'tricks of the trade' and practical recommendations – for beginners and experts alike – on what choices to prioritize when training deep chemical language processing models from scratch. We hope that this paper will accelerate the adoption of deep chemical language processing approaches, and spark novel research to further their potential.

2 Methods

2.1 Molecular string representations

String representations capture two-dimensional molecular information as a sequence of characters ('tokens'). Here, we focus on the two most popular string representations (Fig. 1a and b):

[a]*Eindhoven University of Technology, Institute for Complex Molecular Systems, Eindhoven AI Systems Institute, Dept. Biomedical Engineering, Eindhoven, Netherlands. E-mail: f.grisoni@tue.nl*

[b]*Centre for Living Technologies, Alliance TU/e, WUR, UU, UMC Utrecht, Netherlands*

† Electronic supplementary information (ESI) available. See DOI: https://doi.org/10.1039/d4dd00311j

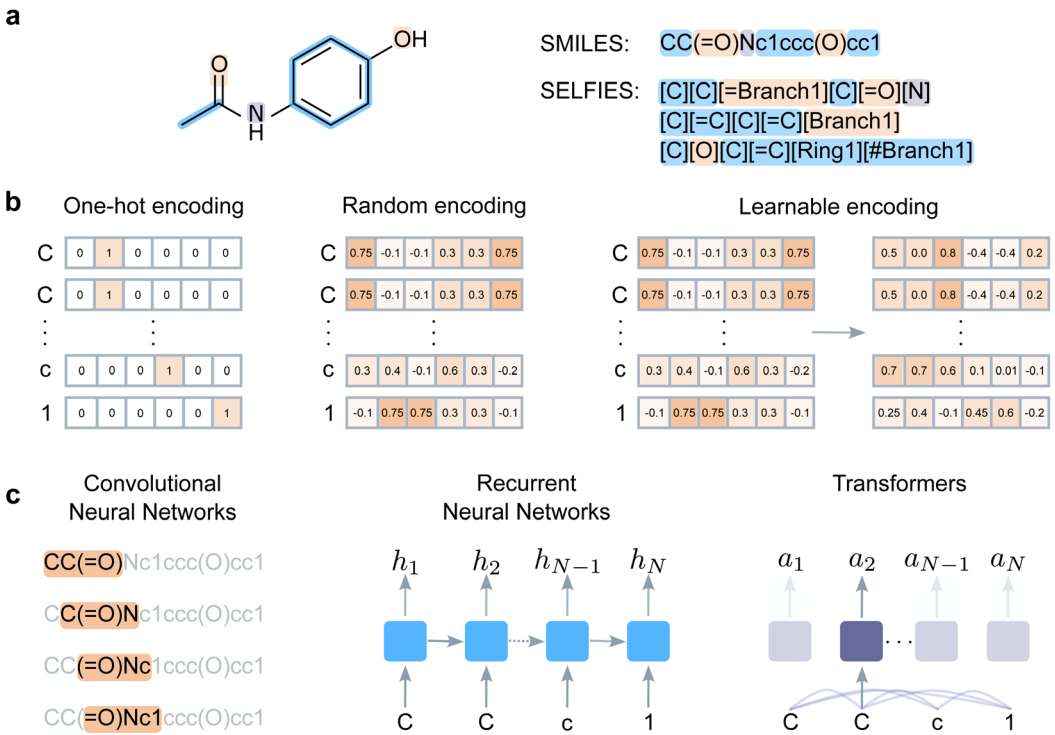


Fig. 1 Deep chemical language processing for bioactivity prediction. (a) String notations such as SMILES and SELFIES represent a molecular graph as a sequence of characters ('tokens'). The atoms are represented with periodic table symbols, while branches, rings, and bonds are assigned special characters. (b) Token encoding, where the chosen molecular string is converted into a matrix to train deep learning models. One-hot encoding represents each token with a unique binary vector. Random encoding maps tokens to fixed, unique, and continuous vectors. Learnable encoding starts with a random vector per token and updates the vectors during training to improve the model performance. (c) Architectures used in this study. Convolutional neural networks slide windows over the input sequences, and learn to weight and aggregate the input elements. Recurrent neural networks iterate over the input tokens in a step-wise manner, and update their 'state' (h_i) to store the information from the previous steps. Transformers learn all-pair relationships between the input tokens and learn to weight each input representation to create the representations in the next layers (a_i).

- Simplified Molecule Input Line Entry Systems (SMILES)[8] strings, which start from any non-hydrogen atom in the molecule and traverse the molecular graph. Atoms are annotated as their element symbols, bonds (except for single bonds) are annotated with special tokens (*e.g.* '=': double, '#': triple), and branching is indicated by bracket opening and closure. Stereochemical information can also be indicated by dedicated tokens, although this will be not considered in this study. Initially proposed for chemical information storage, SMILES strings constitute, to date, the standard notation in chemical language processing.[19,20,25–27]
- Self-Referencing Embedded Strings (SELFIES),[9] which were recently proposed as SMILES alternatives. SELFIES encode the atoms with their symbols, and annotate their connectivity *via* branch length, ring size, and by referencing previous elements. SELFIES strings have been developed for *de novo* design, to mitigate the generation of invalid molecular strings,[9,28–30] and are finding increasing applications for bioactivity prediction.[31,32]

2.2 Token encoding

For deep learning purposes, molecular strings are converted into sequences of vectors, by 'vectorizing' each token in the string. Here, we experimented with three encoding approaches (Fig. 1b), namely:

- One-hot encoding, which represents tokens with V-dimensional binary vectors, V being the number of unique tokens ('vocabulary' size). Each token is allocated a different dimension in this space and has a vector on which only that dimension is set to 1, and the rest is set to 0. One-hot encoding ensures that all token vectors are orthogonal to each other, *i.e.*, the similarity between all tokens is zero.
- Learnable embeddings, whereby a random continuous vector is assigned to each token. These vectors are updated ('learned') during training to optimize the predictions. The updates might enable models to learn relationships between parts of the molecules (and the corresponding tokens) that can be useful for bioactivity prediction. This (and similar) encoding approaches might be referred to as "word embedding"[33] in natural language processing. Here, we use the expression "learnable" to emphasize that these embeddings are updated during training, unlike one-hot and random encoding.
- Random encoding, which assigns a randomly generated continuous vector to each token and uses the same vector throughout the model training. This approach is

intermediate between learnable embeddings and one-hot encoding. Like learnable embeddings, the vectors have continuous values, and they are fixed during training like one-hot encoding.

2.3 Deep learning architectures

We experimented with three well-established deep learning architectures (Fig. 1c). They differ in how they process and combine information on the (encoded) input molecular strings to predict bioactivity.

- Convolutional neural networks (CNNs).[34] CNNs slide windows (called kernels) over an input sequence, and learn to weight input elements at each window. Such window sliding enables CNNs to capture local patterns in sequences, which are then stacked to predict the global properties of a string (*e.g.*, bioactivity).
- Recurrent neural networks (RNNs).[35] RNNs are recurrent models, *i.e.*, they iterate over the input token and, at each step, compress the information into a 'hidden state'. Here, we used bidirectional RNNs – which iterate over the sequence in both directions and concatenate the final hidden states to encode the sequence – and gated recurrent units as the cell type.[36]
- Transformers.[37] Transformers learn patterns between pairs of input tokens, using a mechanism called 'self-attention'. Self-attention learns to represent input sequences by learning to weight the link between every token pair. Since self-attention makes transformers invariant to the token position in the sequence, here we adopted learnable positional embeddings to capture the sequence structure.

2.4 Bioactivity datasets

We curated ten bioactivity datasets containing 1453 to 5500 molecules (Table 1), and spanning two tasks, namely (a) classification (5 datasets), *i.e.*, predicting whether a molecule is active or inactive on a given target (in the form of a label), and (b) regression (5 datasets), where the coefficient of inhibition (K_i) is to be predicted.

- Classification datasets. Five datasets were curated from ExCAPE-DB,[38] which collects ligand-target bioactivity information (in the form of 'active'/'inactive') on 1677 proteins. In this work, we randomly selected five targets: dopamine receptor D3 (DRD3), Flap structure-specific endonuclease 1 (FEN1), mitogen-activated protein kinase kinase kinase kinase 2 (MAP4K2), peptidyl-prolyl *cis*/*trans* isomerase (PIN1), and vitamin D receptor (VDR). For each macromolecular target, a set of 5500 molecules (with 10% of actives) were selected (see Section 2.5).
- Regression. We randomly selected five bioactivity datasets from MoleculeACE,[22] which is based on ChEMBL.[39] The following datasets were used for pK_i prediction: serotonin 1a receptor (5-HT1A), μ-opioid receptor 1 (MOR), dopamine receptor D3 (DRD3), sigma opioid receptor 1 (SOR), and serine/threonine-protein kinase PIM1 (PIM1). These datasets were selected to span several target families and to ensure a sufficient number of molecules available for training and testing (from 1453 to 2596).

The classification datasets have more molecules than the regression datasets and were built to contain structurally diverse molecules (see Section 2.5). Hence, they can be seen as a proxy for hit discovery campaigns, where structurally novel, and bioactive molecules are searched for. Conversely, the regression datasets, which originate from ChEMBL, mostly contain series of highly similar molecules, hence resembling a lead optimization campaign. The selected datasets include five receptors and four enzymes, and they span several families – such as G protein-coupled receptors, nuclear receptors, and kinases. Such diversity ensures that our analysis covers a broad range of targets for drug discovery.

Table 1 Datasets used in this study. We curated ten bioactivity datasets, for classification (*i.e.*, binding *vs.* non-binding[38]) and regression (*i.e.*, pK_i prediction[22]) purposes. For each dataset, we report ID, target name, and total number of molecules (n)

Task	ID	Target name	n
Class.	DRD3	Dopamine receptor D3	5500
	FEN1	Flap structure-specific endonuclease 1	5500
	MAP4K2	Mitogen-activated protein 4× kinase 2	5500
	PIN1	Peptidyl-prolyl *cis*/*trans* isomerase	5500
	VDR	Vitamin D receptor	5500
Reg.	5-HT1A	Serotonin 1a receptor	3298
	MOR	μ-Opioid receptor	2838
	DRD3	Dopamine receptor D3	3596
	SOR	Sigma opioid receptor	1325
	PIM1	Serine/threonine-protein kinase PIM1	1453

2.5 Experimental setup

2.5.1 Data preparation

- Classification. We randomly sampled 350 actives and 3500 inactives from ExCAPE-DB as the training set for each selected target. The validation set was curated by selecting 75 actives and 750 inactives that were distant from the training set. This was quantified by both (a) a minimum edit distance on canonical SMILES strings larger than 10, and (b) a maximum Tanimoto similarity on extended connectivity fingerprints[40] smaller than 60%. The test set was collected the same way and it contains 75 actives and 750 inactives that are dissimilar both to training and validation molecules.
- Regression. For each target, we created five folds of training, validation, and test sets (70%, 15%, 15%, respectively). We heuristically minimized train-test similarity by first grouping molecules based on substructure similarity, and then dividing them into training and test set (*via* deepchem, FingerprintSplitter[41]).

For all collected molecules, we removed stereochemistry, sanitized the molecules, and canonicalized the SMILES strings (rdkit v2020.09.01). We filtered out the molecules with canonical SMILES strings longer than 75 tokens; created the SELFIES strings for all retained molecules; and applied padding to the maximum sequence length. Our data curation pipeline led to different distributions of molecular similarities between

Table 2 Model hyperparameters. Grid search is used to optimize model hyperparameters

Model	Hyperparameter	Search space
All	No. layers	1, 2, 3
	Dropout	0.25
	Batch size	32
CNN	No. filters	32, 64, 128
	Kernel length	3, 5, 7
	Learning rate	10^{-3}, 5×10^{-3}, 10^{-4}, 5×10^{-5}
RNN	Hidden state dim.	16, 32, 64, 128
	Learning rate	10^{-2}, 10^{-3}, 5×10^{-3}, 10^{-4}, 5×10^{-5}
Transformer	No. heads	1, 2, 4
	MLP dim.	32, 64, 128
	Learning rate	10^{-3}, 5×10^{-3}, 10^{-4}, 5×10^{-5}
XGBoost (baseline)	No. trees	2000
	Max. depth	3, 4, 5
	Eta	0.01, 0.05, 0.1, 0.2
	Column fraction	0.5, 0.75, 1.0
	Sample fraction	0.5, 0.75, 1.0

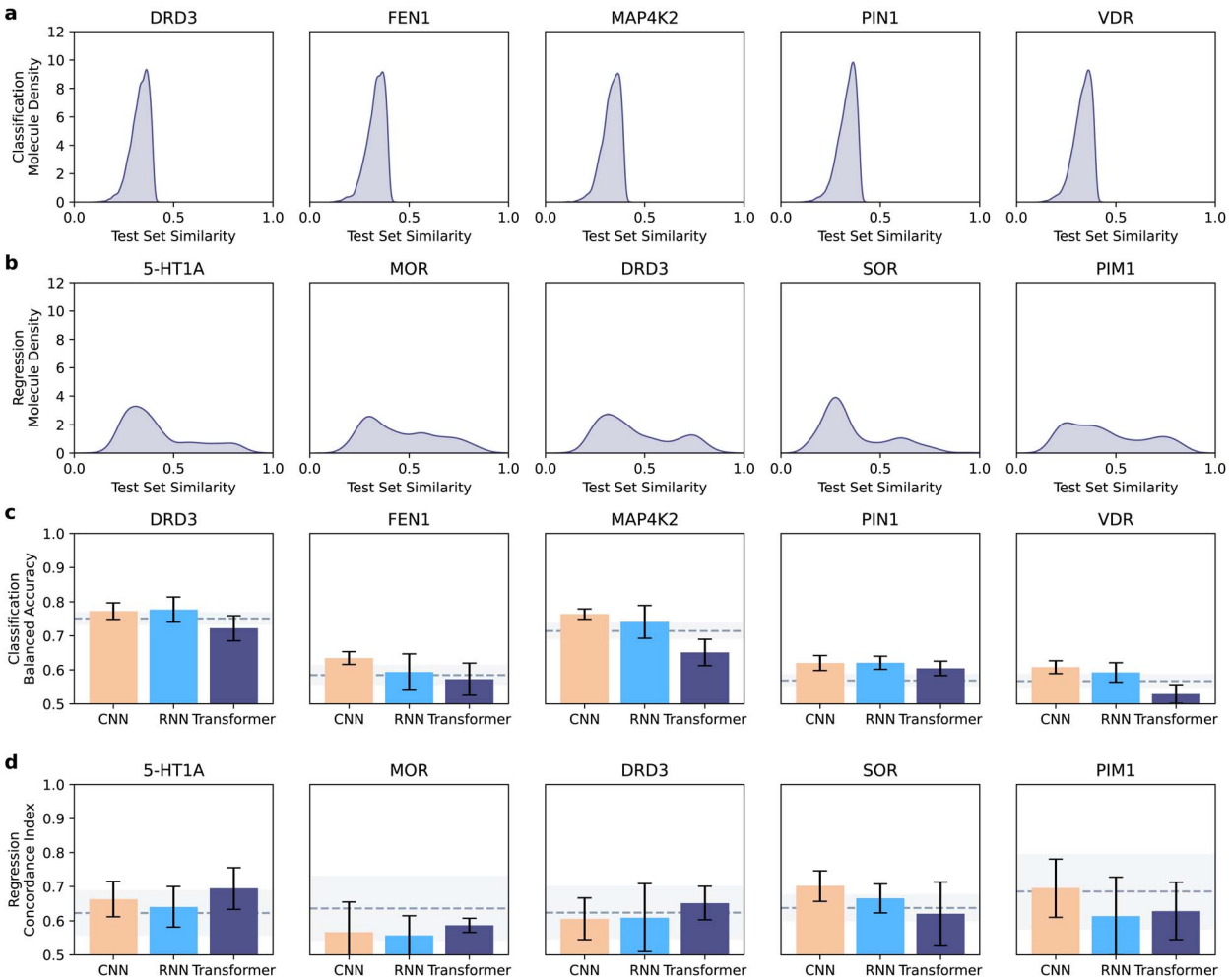


Fig. 2 Overview of dataset similarity and of model performance. (a and b) Distribution of test set similarities in comparison with training set molecules. The similarity was quantified as the Tanimoto coefficient on extended connectivity fingerprints,[40] and the maximum similarity was reported. Different distributions can be observed in the classification (a) and regression (b) datasets, with the former containing less similar molecules on average. (c and d) Performance of neural network architectures across datasets. Bar plots indicate the mean test set performance (with error bars denoting the standard deviation), in comparison with the XGBoost baseline (dashed line: average performance, shaded area: standard deviation). Performance was quantified as balanced accuracy in classification (c), and as concordance index in regression (d).

training and test set molecules for classification (Fig. 2a) and regression datasets (Fig. 2b).

2.5.2 Model training and optimization. We tested all combinations of (a) model architectures (CNN, RNN, and Transformers), (b) molecular strings (SMILES and SELFIES), and encoding approaches (one-hot, random, and learnable) for all datasets.[42] We optimized hyperparameters for each combination and each dataset separately (Table 2) using five-fold Monte Carlo validation.[43] A three-layer perceptron was used as a prediction module for consistency. Finally, XGBoost models[44] were trained on extended connectivity fingerprints[40] as baselines across all datasets. Early stopping with a patience of five epochs (or trees for XGBoost) and a tolerance of 10^{-5} on validation loss were used. For classification models, we used loss re-weighting to tackle the data imbalance, which assigns the inverted frequency of classes as weights to molecules during loss computation. Finally, the best models were selected based on validation loss, *i.e.*, cross-entropy and mean square error for classification and regression, respectively.

2.6 Performance evaluation

The performance of classification models was evaluated *via* the balanced accuracy (BA), expressed as follows:

$$\mathrm{BA} = \frac{1}{2}\left(\frac{\mathrm{TP}}{n\mathrm{P}} + \frac{\mathrm{TN}}{n\mathrm{N}}\right), \quad (1)$$

where TP and TN are the numbers of correctly classified positives and negatives, while nP and nN are the total number of positive and negative molecules, respectively.

The performance of regression models was evaluated *via* concordance index,[45,46] which quantifies the model's ability to rank molecules by their experimental potency based on the predicted potency. Both metrics are bound between 0 and 1 – the closer to 1, the better the performance. Additional classification (accuracy, precision, recall, and $F1$ score) and regression metrics (root mean square error and R^2) can be found as ESI† and in the GitHub repository.

3 Results

3.1 Choosing a neural network architecture

Here, we aim to gather insights into the effect of the model architecture (CNN *vs.* RNN *vs.* Transformers) on the performance. To this end, we analyzed the best models per architecture (chosen on the validation set, and analyzed on the test set), regardless of the molecule representation and encoding strategies (Fig. 2c and d).

CNNs were the best-performing approach in classification for three targets (FEN1, MAP4K2, and VDR), and RNNs achieved the highest balanced accuracy in the other two (DRD3 and PIN1). Similar results are obtained when observing the ability of each model to recognize positive molecules (*e.g.*, *via* recall, ESI Fig. 1a†). In regression, CNNs outperformed the other approaches on two out of five targets (SOR and PIN1) and transformers yielded the top-performing models on three out of five targets (5-HT1A, MOR, and DRD3). When looking at the model error (*via* root mean squared error), similar trends are observable, although with minor, dataset-dependent differences (ESI Fig. 1b†). A Wilcoxon signed-ranked test on pooled balanced accuracies and concordance indices per task across targets (after a Friedman test and Holm–Bonferroni p-value correction) indicated that CNN is the best CLP architecture in classification (corrected $p = 7 \times 10^{-2}$ and $p = 6 \times 10^{-6}$, against RNN and Transformer, respectively; ESI Tables 1 and 2†). No statistically significant difference was observed between architectures in the regression cases (using the same statistical procedure).

Interestingly, CNN outperformed the XGBoost baseline in all classification datasets, where the test set molecules are structurally dissimilar to the training set (Tanimoto similarity on extended connectivity fingerprints lower than 0.4, Fig. 2a). In regression, where the test set molecules are more similar to the training set (Fig. 2b), neither deep models nor XGBoost is statistically superior across the datasets per the same statistical procedure as before ($\alpha = 0.05$), including the larger datasets where deep learning models might be expected to perform better.[5] These results suggest that CLP approaches, and in particular, CNNs might have a higher potential than 'traditional' machine learning models when applied to molecules that are structurally diverse from the training set.

Hence, when considering their performance, architectural simplicity (compared to transformers) and training speed (compared to RNNs), convolutional neural networks constitute the ideal starting choice for chemical language processing and bioactivity prediction.

3.2 Representing and encoding molecular structures

Here, we aimed to unveil the effect of the chosen molecular string representation (SMILES *vs.* SELFIES) and token embedding (one-hot, random, and learnable) strategies. To this end, we compared the best models for each molecule representation and token encoding (minimum average error on the validation set). When investigating for practical guidelines, the differences are less evident than when choosing a neural network architecture (Fig. 3).

SMILES strings yield higher performance than SELFIES across classification tasks ($p < 0.05$, Wilcoxon signed-ranked test). In regression, SELFIES outperformed SMILES strings on two datasets (DRD3 and PIM1), and showed similar performance otherwise, without statistically significant differences (Wilcoxon signed-ranked test, $\alpha = 0.05$). Similar trends are observed when measuring recall and concordance index (ESI Fig. 2†) Neither SMILES nor SELFIES strings consistently stand out in smaller or larger datasets, indicating that they might capture a similar type and amount of information. Overall, the performance differences due to the chosen string notation were lower than those caused by the model architecture, where the differences were statistically significant.

When analyzing the encoding strategies, no approach consistently outperformed the others (Fig. 3c and d and ESI Fig. 3†), suggesting that all encoding approaches impact bioactivity prediction comparably. This underscores that, in the space of our design of experiments, choosing model

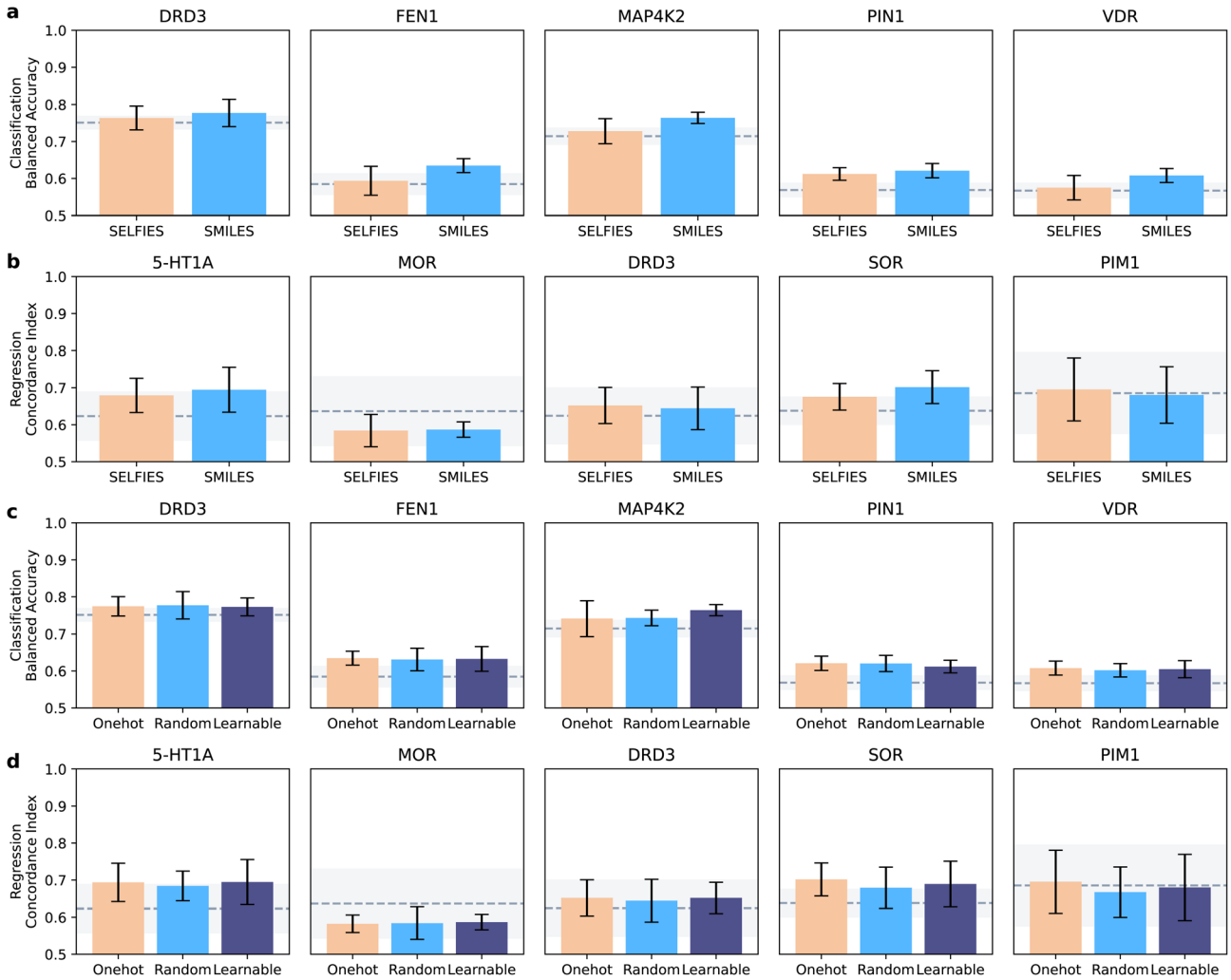


Fig. 3 Effect of input molecular strings and of token encoding strategies. (a and b) Performance of SMILES and SELFIES representations on the model performance. Classification (a) and regression dataset (b) are analyzed separately. (c and d) Performance of token encoding strategies on classification (c) and regression (d). For all plots, bars indicate the mean performance on the test set of each notation, and error bars indicate the standard deviation. The performance of the XGBoost baseline is also indicated (dashed line: average; shaded area: standard deviation).

architecture first, and then molecular string notations, should have higher priority than the encoding strategy.

When considering these results, we recommend CLP hitchhikers[42] to use SMILES strings combined with learnable encoding. SMILES strings are, in fact, ubiquitous in available databases, and numerous tools exist to process them (*e.g.*, rdkit). This aspect makes SMILES strings easier to work with, with no loss in performance. Learnable representations are also

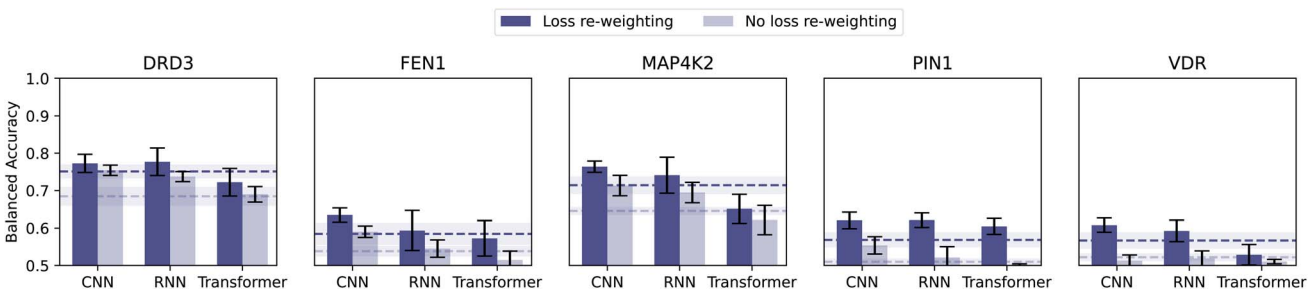


Fig. 4 Effect of loss re-weighting. Comparison of the classification performance obtained with and without loss re-weighting (*i.e.*, assigning different weights to the molecules, as the inverse of their class frequency).

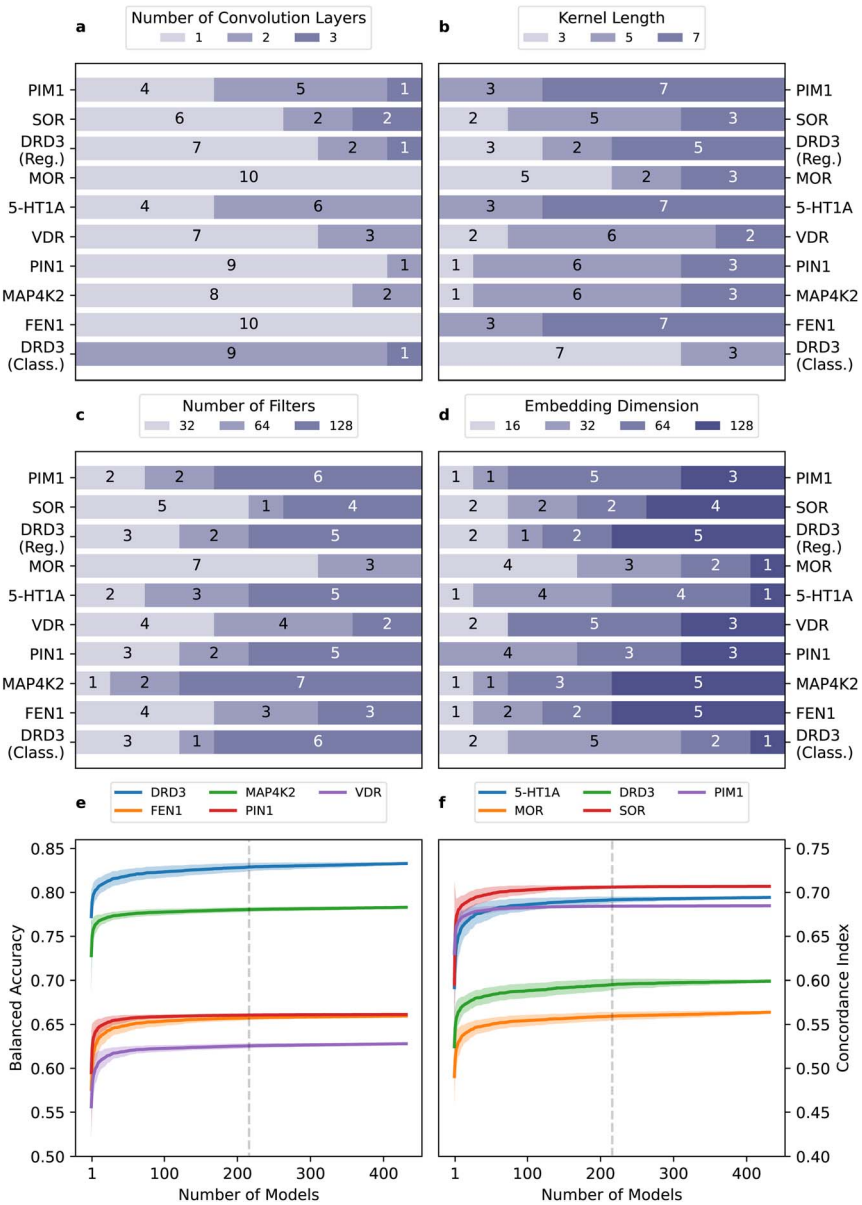


Fig. 5 Hyperparameter tuning. (a–d) Most frequently occurring hyperparameter values among the top-ten models per dataset (CNN architecture, with SMILES strings and learnable embeddings). The following parameters were investigated: number of convolution layers (a), kernel length (b), number of filters (c), and token embedding dimension (d). (e and f) Model performance *vs.* explored hyperparameter space size. Performance of progressively subsampled models from 1 to 432 hyperparameter configurations (total) for both classification (e) and regression (f). The dashed line indicates 50% of models being explored.

simple to use, and are implemented in most major deep learning libraries (*e.g.*, Pytorch,[47] Tensorflow,[48] and Keras[49]).

3.3 Other tricks of the trade

While the previous sections have tackled the most important algorithm-design choices in CLP, there are still many 'bells and whistles'[24] involved in obtaining predictive models. In what follows, we will focus on the loss function and hyperparameter optimization – both aspects impacting the effectiveness of the training process, and, ultimately, the model predictivity.

3.3.1 Loss functions for imbalanced classes. Class imbalance is common in bioactivity datasets,[50] since desirable outcomes (*e.g.*, bioactive or non-toxic molecules) occur less frequently. Hence, mitigating the negative effects of class imbalance on the model performance is key for CLP hitchhikers.[42]

To mitigate class imbalance, in all the classification results shown so far, we applied loss re-weighting. We assigned a weight of 10 to the active molecules and of 1 to the inactive molecules (corresponding to the inverse of their respective class frequency). Here we train models with no loss adjustment

(using the same experimental setup as before) and quantify the impact.

Loss re-weighting substantially increased balanced accuracy, 6% on average (Fig. 4). In some extreme cases (*i.e.*, FEN1, PIN1, and VDR), equal loss weighting dropped the balanced accuracy to 0.5 (baseline-level performance). Loss re-weighting is hence a simple and effective strategy that we recommend to mitigate class imbalance, among other options.[51]

3.3.2 Optimal hyperparameters. Hyperparameter optimization can be a demanding task due to the high number of hyperparameters to explore and required domain expertise. To equip CLP practitioners with guidelines, we focused on our recommended setup (CNNs trained on SMILES strings with learnable embedding), and inspected the top-10 performing models (the test set average) for the following hyperparameters (Fig. 5): (a) number of convolution layers, impacting the network depth and complexity, (b) kernel length, controlling the size of learned patterns, (c) number of convolution filters, controlling information compression across layers, and (d) token embedding dimension, controlling the size of the latent representations learned.

The best-performing models tend to have a low number of layers, with one being the most prevalent (seven out of ten datasets, and 65% occurrence, (Fig. 5a). Optimal kernel size and number of filters (Fig. 5b and c) results are dataset dependent. Finally, embeddings of 32 or higher dimensions are preferred (84% of cases (Fig. 5d). These results offer indications for hyperparameter prioritization 'on a budget', although we recommend conducting extensive searches whenever feasible.

3.3.3 Exploring the hyperparameter galaxy. To provide guidelines for parsimonious hyperparameter optimization, we randomly sampled an increasing number of models from the hyperparameter space (from 1 to the total, 432), and analyzed the performance of the top ten models (Fig. 5e and f). Performance often plateaued before reaching 100 models, with a shrink in its variability when half of the space was explored. These findings indicate that defining a high-dimensional hyperparameter space can be better than relying on a narrow one, and that randomly exploring half of the grid can be sufficient to reach the maximum performance level possible in that space.

4 So long, and thanks for all the data

Casting molecular tasks as chemical language processing has achieved enormous success in the molecular sciences,[13,19] owed to a unique combination of simplicity (*e.g.*, in representing and processing molecules as strings) and performance.[52,53] The importance of chemical language processing is hence only expected to increase. To accelerate the adoption of CLP approaches by novices and experts alike, these are our guidelines for hitchhikers,[42] based on the data we have collected:

4.1 'KISS: Keep It Simple, Silly!'

Convolutional neural networks – an architecture that is simpler than the Transformer and faster than recurrent neural networks – yielded the best performance overall, and are recommended as the first choice. Since representation and encoding strategies minimally affected performance, we recommend using SMILES strings for their ubiquity in databases and software, and learnable embeddings for existing implementations in most deep learning packages. Combining various architectures and representations could enhance performance, though it may require larger datasets to support the added complexity.

4.2 'Cut your losses'

Molecular bioactivity datasets are inherently imbalanced,[50] and the 'losses' due to such imbalance should be minimized to ensure predictivity.[54] We recommend loss re-weighting as a simple and yet effective strategy to increase model performance.

4.3 'Cast a wide fishing net'

Hyperparameter optimization can be computationally demanding. Here, we show that, in general, networks with a low (one to two) number of layers tend to perform well enough, while other hyperparameter choices depend on the dataset. In general, once a hyperparameter space is defined, optimal hyperparameters are likely to be found by exploring half of the possible combinations. Hence, we recommend casting a broad (rather than a narrow) hyperparameter grid for exploration, and refine the hyperparameter values at a later stage.

Several other fascinating properties of the 'chemical language' can further the potential of CLP approaches. One of them is molecular string augmentation,[12] where multiple molecular strings can be used to represent the same molecule, *e.g.*, to increase the number of data available for training,[55,56] or for uncertainty estimation.[20,57] Moreover, transfer learning[58] can be particularly effective on molecular strings,[21,59] *e.g.*, to mitigate the limited data availability on a specific target. Our hitchhiker's guide explores various protein families, within which no consistent trends are observed. However, the utility of our analysis could be further enhanced by expanding the datasets, *e.g.*, by incorporating additional families or further exploring existing ones. This would further improve the generalization of our conclusions. We encourage 'CLP hitchhikers' to venture forth into such elements and assess their effectiveness on a case-by-case basis.

Data availability

All the code and data useful to reproduce the results of this study are available on GitHub at the following URL: **https://github.com/molML/chemical-language-processing-for-bioactivity-prediction**. The code and data at the time of publishing are available on Zenodo at: **https://doi.org/10.5281/zenodo.14423621**.

Author contributions

Conceptualization: both authors. Data curation: RÖ. Formal analysis: both authors. Investigation: both authors. Methodology: both authors. Software: RÖ. Visualization: RÖ. Writing – original draft: RÖ. Writing – review and editing: both authors.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

This research was co-funded by the European Union (ERC, ReMINDER, 101077879). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. The authors also acknowledge support from the Irene Curie Fellowship and the Centre for Living Technologies.

Notes and references

1 J. Vamathevan, D. Clark, P. Czodrowski, I. Dunham, E. Ferran, G. Lee, B. Li, A. Madabhushi, P. Shah, M. Spitzer, *et al.*, *Nat. Rev. Drug Discovery*, 2019, **18**, 463–477.
2 R. Özçelik, D. van Tilborg, J. Jiménez-Luna and F. Grisoni, *ChemBioChem*, 2023, **24**, e202200776.
3 R. Chakraborty and Y. Hasija, *Expert Syst. Appl.*, 2023, **229**, 120592.
4 J. M. Stokes, K. Yang, K. Swanson, W. Jin, A. Cubillos-Ruiz, N. M. Donghia, C. R. MacNair, S. French, L. A. Carfrae, Z. Bloom-Ackermann, *et al.*, *Cell*, 2020, **180**, 688–702.
5 D. van Tilborg, H. Brinkmann, E. Criscuolo, L. Rossen, R. Özçelik and F. Grisoni, *Curr. Opin. Struct. Biol.*, 2024, **86**, 102818.
6 O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel and T. Langer, *Drug Discovery Today: Technol.*, 2020, **37**, 1–12.
7 X. Zeng, S.-J. Li, S.-Q. Lv, M.-L. Wen and Y. Li, *Front. Pharmacol.*, 2024, **15**, 1375522.
8 D. Weininger, *J. Chem. Inf. Comput. Sci.*, 1988, **28**, 31–36.
9 M. Krenn, F. Häse, A. Nigam, P. Friederich and A. Aspuru-Guzik, *Mach. Learn.: Sci. Technol.*, 2020, **1**, 045024.
10 H. Öztürk, A. Özgür and E. Ozkirimli, *Bioinformatics*, 2018, **34**, i821–i829.
11 Q. Zhao, G. Duan, M. Yang, Z. Cheng, Y. Li and J. Wang, *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2022, **20**, 852–863.
12 E. J. Bjerrum, *arXiv*, 2017, preprint, arXiv:1703.07076, DOI: **10.48550/arXiv.1703.07076**.
13 H. Öztürk, A. Özgür, P. Schwaller, T. Laino and E. Ozkirimli, *Drug Discovery Today*, 2020, **25**, 689–705.
14 J. Ross, B. Belgodere, V. Chenthamarakshan, I. Padhi, Y. Mroueh and P. Das, *Nat. Mach. Intell.*, 2022, **4**, 1256–1264.
15 N. O'Boyle and A. Dalke, *ChemRxiv*, 2018, DOI: **10.26434/chemrxiv.7097960.v1**.
16 J.-N. Wu, T. Wang, Y. Chen, L.-J. Tang, H.-L. Wu and R.-Q. Yu, *Nat. Commun.*, 2024, **15**, 4993.
17 S. R. Heller, A. McNaught, I. Pletnev, S. Stein and D. Tchekhovskoi, *J. Cheminf.*, 2015, **7**, 1–34.
18 E. Noutahi, C. Gabellini, M. Craig, J. S. Lim and P. Tossou, *Digital Discovery*, 2024, **3**, 796–804.
19 F. Grisoni, *Curr. Opin. Struct. Biol.*, 2023, **79**, 102527.
20 T. B. Kimber, M. Gagnebin and A. Volkamer, *Artif. Intell. Life Sci.*, 2021, **1**, 100014.
21 M. Moret, I. Pachon Angona, L. Cotos, S. Yan, K. Atz, C. Brunner, M. Baumgartner, F. Grisoni and G. Schneider, *Nat. Commun.*, 2023, **14**, 114.
22 D. van Tilborg, A. Alenicheva and F. Grisoni, *J. Chem. Inf. Model.*, 2022, **62**, 5938–5951.
23 Y. Zhou, S. Cahya, S. A. Combs, C. A. Nicolaou, J. Wang, P. V. Desai and J. Shen, *J. Chem. Inf. Model.*, 2018, **59**, 1005–1016.
24 Y. Bengio, *Neural networks: Tricks of the trade*, Springer, 2nd edn, 2012, pp. 437–478.
25 R. Özçelik, H. Öztürk, A. Özgür and E. Ozkirimli, *Mol. Inf.*, 2021, **40**, 2000212.
26 A. Sharma, R. Kumar, S. Ranjta and P. K. Varadwaj, *J. Chem. Inf. Model.*, 2021, **61**, 676–688.
27 C.-K. Wu, X.-C. Zhang, Z.-J. Yang, A.-P. Lu, T.-J. Hou and D.-S. Cao, *Briefings Bioinf.*, 2021, **22**, bbab327.
28 A. Nigam, R. Pollice, M. Krenn, G. dos Passos Gomes and A. Aspuru-Guzik, *Chem. Sci.*, 2021, **12**, 7079–7090.
29 J. Choi, S. Seo, S. Choi, S. Piao, C. Park, S. J. Ryu, B. J. Kim and S. Park, *Comput. Biol. Med.*, 2023, **157**, 106721.
30 M. Krenn, Q. Ai, S. Barthel, N. Carson, A. Frei, N. C. Frey, P. Friederich, T. Gaudin, A. A. Gayle, K. M. Jablonka, *et al.*, *Patterns*, 2022, **3**, 100588.
31 A. Yüksel, E. Ulusoy, A. Ünlü and T. Doğan, *Mach. Learn.: Sci. Technol.*, 2023, **4**, 025035.
32 Y. Feng, Y. Zhang, Z. Deng and M. Xiong, *Quant. Biol.*, 2024, 141–154.
33 Y. Bengio, R. Ducharme and P. Vincent, *Advances in Neural Information Processing Systems*, 2000, vol. 13, **https://papers.nips.cc/paper_files/paper/2000/hash/728f206c2a01bf572b5940d7d9a8fa4c-Abstract.html**.
34 Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, *Proc. IEEE*, 1998, **86**, 2278–2324.
35 J. J. Hopfield, *Proc. Natl. Acad. Sci. U. S. A.*, 1982, **79**, 2554–2558.
36 K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk and Y. Bengio, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1724–1734.
37 A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, *Advances in Neural Information Processing Systems*, 2017, vol. 30, **https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html**.
38 J. Sun, N. Jeliazkova, V. Chupakhin, J.-F. Golib-Dzib, O. Engkvist, L. Carlsson, J. Wegner, H. Ceulemans, I. Georgiev, V. Jeliazkov, *et al.*, *J. Cheminf.*, 2017, **9**, 1–9.

39 A. Gaulton, A. Hersey, M. Nowotka, A. P. Bento, J. Chambers, D. Mendez, P. Mutowo, F. Atkinson, L. J. Bellis, E. Cibrián-Uhalte, *et al.*, *Nucleic Acids Res.*, 2017, **45**, D945–D954.
40 D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.
41 B. Ramsundar, P. Eastman, P. Walters, V. Pande, K. Leswing and Z. Wu, *Deep Learning for the Life Sciences*, O'Reilly Media, 2019.
42 The Answer to the Ultimate Question of Life, the Universe, and Everything.
43 Q.-S. Xu and Y.-Z. Liang, *Chemom. Intell. Lab. Syst.*, 2001, **56**, 1–11.
44 T. Chen and C. Guestrin, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.
45 M. Gönen and G. Heller, *Biometrika*, 2005, **92**, 965–970.
46 T. Pahikkala, A. Airola, S. Pietilä, S. Shakyawar, A. Szwajda, J. Tang and T. Aittokallio, *Briefings Bioinf.*, 2014, bbu010.
47 A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein and L. Antiga, *et al.*, *Advances in Neural Information Processing Systems*, 2019, vol. 32, **https://papers.nips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html**.
48 M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu and X. Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015, software available from **https://www.tensorflow.org/**.
49 F. Chollet, *Keras*, **https://github.com/fchollet/keras**, 2015.
50 A. Volkamer, S. Riniker, E. Nittinger, J. Lanini, F. Grisoni, E. Evertsson, R. Rodríguez-Pérez and N. Schneider, *Artif. Intell. Life Sci.*, 2023, **3**, 100056.
51 Q. Wang, Y. Ma, K. Zhao and Y. Tian, *Ann. Data Sci.*, 2020, 1–26.
52 D. Flam-Shepherd, K. Zhu and A. Aspuru-Guzik, *Nat. Commun.*, 2022, **13**, 3293.
53 H. Öztürk, E. Ozkirimli and A. Özgür, *BMC Bioinf.*, 2016, **17**, 1–11.
54 A. Fernández, S. García, M. Galar, R. C. Prati, B. Krawczyk and F. Herrera, *Learning from imbalanced data sets*, Springer, 2018, vol. 10.
55 C. Li, J. Feng, S. Liu and J. Yao, *Computational Intelligence and Neuroscience*, 2022, **2022**, 8464452.
56 T. B. Kimber, S. Engelke, I. V. Tetko, E. Bruno and G. Godin, *arXiv*, 2018, preprint, arXiv:1812.04439, DOI: **10.48550/arXiv.1812.04439**.
57 R. Birolo, R. Özçelik, A. Aramini, R. Gobetto, M. R. Chierotti and F. Grisoni, *ChemRxiv*, 2024, preprint, DOI: **10.26434/chemrxiv-2024-vgvhk-v3**.
58 C. Cai, S. Wang, Y. Xu, W. Zhang, K. Tang, Q. Ouyang, L. Lai and J. Pei, *J. Med. Chem.*, 2020, **63**, 8683–8694.
59 G. Uludoğan, E. Ozkirimli, K. O. Ulgen, N. Karalı and A. Özgür, *Bioinformatics*, 2022, **38**, ii155–ii161.