

Université de Montréal

**Reduced Collision Fingerprints and Pairwise Molecular
Comparisons for Explainable Property Prediction
Using Deep Learning**

par

Thomas MacDougall

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

Orientation Intelligence Artificielle

16 Août 2021

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

**Reduced Collision Fingerprints and Pairwise
Molecular Comparisons for Explainable
Property Prediction Using Deep Learning**

présenté par

Thomas MacDougall

a été évalué par un jury composé des personnes suivantes :

Dr. Gauthier Gidel

(président-rapporteur)

Dr. Sébastien Lemieux

(directeur de recherche)

Dr. Anne Marinier

(codirecteur)

Dr. Bang Lui

(membre du jury)

Résumé

Les relations entre la structure des composés chimiques et leurs propriétés sont complexes et à haute dimension. Dans le processus de développement de médicaments, plusieurs propriétés d'un composé doivent souvent être optimisées simultanément, ce qui complique encore la tâche. Ce travail explore deux représentations des composés chimiques pour les tâches de prédiction des propriétés. L'objectif de ces représentations proposées est d'améliorer l'explicabilité afin de faciliter le processus d'optimisation des propriétés des composés. Premièrement, nous décomposons l'algorithme ECFP (Extended connectivity Fingerprint) et le rendons plus simple pour la compréhension humaine. Nous remplaçons une fonction de hachage sujet aux collisions par une relation univoque de sous structure à bit. Nous constatons que ce changement ne se traduit pas par une meilleure performance prédictive d'un perceptron multicouche par rapport à l'ECFP. Toutefois, si la capacité du prédicteur est ramenée à celle d'un prédicteur linéaire, ses performances sont meilleures que celles de l'ECFP. Deuxièmement, nous appliquons l'apprentissage automatique à l'analyse des paires moléculaires appariées (MMPA), un paradigme de conception du développement de médicaments. La MMPA compare des paires de composés très similaires, dont la structure diffère par une modification sur un site. Nous formons des modèles de prédiction sur des paires de composés afin de prédire les différences d'activité. Nous utilisons des contraintes de similarité par paires comme MMPA, mais nous utilisons également des paires échantillonnées de façon aléatoire pour entraîner les modèles. Nous constatons que les modèles sont plus performants sur des paires choisies au hasard que sur des paires avec des contraintes de similarité strictes. Cependant, les meilleurs modèles par paires ne sont pas capables de battre les performances de prédiction du modèle simple de base. Ces deux études, RCFP et comparaisons par paires, visent à aborder la prédiction des propriétés d'une manière plus compréhensible. En utilisant l'intuition et l'expérience des chimistes médicaux dans le cadre de la modélisation prédictive, nous espérons encourager l'explicabilité en tant que composante nécessaire des modèles chiminformatiques prédictifs.

Mots Clés: Chimie Médicinale, Développement de médicaments, Apprentissage automatique, Chimie-informatique, Apprentissage profond, Prédiction de la bioactivité, Représentations chimiques

Abstract

The relationships between the structure of chemical compounds and their properties are complex and high dimensional. In the drug development process, multiple properties of a compound often need to be optimized simultaneously, further complicating the task. This work explores two representations of chemical compounds for property prediction tasks. The goal of these suggested representations is improved explainability to better understand the compound property optimization process. First, we decompose the Extended Connectivity Fingerprint (ECFP) algorithm and make it more straightforward for human understanding. We replace a collision-prone hash function with a one-to-one substructure-to-bit relationship. We find that this change which does not translate to higher predictive performance of a multi-layer perceptron compared to ECFP. However, if the capacity of the predictor is lowered to that of a linear predictor, it does perform better than ECFP. Second, we apply machine learning to Matched Molecular Pair Analysis (MMPA), a drug development design paradigm. MMPA compares pairs of highly similar compounds, differing in structure by modification at one site. We train prediction models on pairs of compounds to predict differences in activity. We use pairwise similarity constraints like MMPA, but also use randomly sampled pairs to train the models. We find that models perform better on randomly chosen pairs than on pairs with strict similarity constraints. However, the best pairwise models are not able to beat the prediction performance of the simpler baseline single model. Both of these investigations, RCFP and pairwise comparisons, aim to approach property prediction in a more explainable way. By using intuition and experience of medicinal chemists within predictive modelling, we hope to encourage explainability as a necessary component of predictive cheminformatic models.

Keywords: Medicinal Chemistry, Drug Development, Cheminformatics, Machine Learning, Deep Learning, Graph Convolutional Neural Networks, Bioactivity Prediction, Chemical Representations, Extended Connectivity Fingerprints, Matched Molecular Pair Analysis

Contents

| | |
|---|-------|
| Résumé | v |
| Abstract | vii |
| List of tables | xiii |
| List of figures | xv |
| List of Abbreviations | xix |
| Acknowledgements | xxiii |
| Chapter 1. Introduction | 1 |
| 1.1. Predictive Cheminformatics | 1 |
| 1.1.1. Structure-Activity Relationships | 1 |
| 1.1.2. Properties and Activities | 2 |
| 1.1.3. Quantitative Structure-Activity Relationships | 4 |
| 1.1.4. Ligand-based and Structure-Based Drug Design | 5 |
| 1.1.5. Virtual Screening | 6 |
| 1.2. Chemical Representations | 7 |
| 1.2.1. Making Good Chemical Representations | 7 |
| 1.2.2. SMILES | 8 |
| 1.2.3. Chemical Descriptors | 9 |
| 1.2.4. Fingerprints | 10 |
| 1.2.5. Extended Connectivity Fingerprints (ECFP) | 10 |
| 1.3. Machine Learning in Chemistry | 14 |
| 1.3.1. Machine Learning (ML) | 14 |
| 1.3.2. Early ML in Chemistry | 17 |
| 1.3.3. Early Deep Learning in Chemistry | 18 |
| 1.3.4. Deep Learning in Other Fields | 19 |
| 1.3.5. Message Passing Neural Networks (MPNNs) in Chemistry | 20 |

| | | |
|-------------------|--|-----------|
| 1.4. | Medicinal Chemistry Intuition | 23 |
| 1.4.1. | The Drug Development Pipeline | 24 |
| 1.4.1.1. | The Discovery Phase..... | 24 |
| 1.4.1.2. | The Preclinical Development Phase | 24 |
| 1.4.1.3. | The Clinical Development Phase..... | 24 |
| 1.4.1.4. | The Product Approval and Launch Phase..... | 25 |
| 1.4.2. | Computational Methods In The Pipeline..... | 25 |
| 1.4.3. | Molecular Matched Pair Analysis | 25 |
| 1.5. | Project Goals and Approaches | 27 |
| Chapter 2. | Datasets and Implementation..... | 29 |
| 2.1. | Datasets | 29 |
| 2.1.1. | AurA | 29 |
| 2.1.2. | BACE | 30 |
| 2.1.3. | CEP | 30 |
| 2.2. | Implementation Details | 30 |
| 2.3. | Baseline Predictor | 31 |
| 2.4. | Metrics | 31 |
| 2.5. | Hyperparameters | 32 |
| Chapter 3. | Reduced Collision Fingerprints (RCFP) | 33 |
| 3.1. | Motivation..... | 33 |
| 3.2. | Methodology..... | 35 |
| 3.3. | Results and Discussion..... | 39 |
| 3.3.1. | Similarity Metrics..... | 39 |
| 3.3.2. | <i>m</i> -Cutoff..... | 41 |
| 3.3.3. | Prediction Performance | 42 |
| Chapter 4. | Pairwise Compound Comparisons | 47 |
| 4.1. | Predictor Descriptions | 47 |
| 4.2. | Fingerprint Types..... | 50 |
| 4.2.1. | Fingerprint Type Descriptions | 50 |
| 4.2.1.1. | Extended Connectivity Fingerprints (ECFP) | 50 |

| | | |
|-------------------|---|-----------|
| 4.2.1.2. | Reduced Collision Fingerprints (RCFP)..... | 51 |
| 4.2.1.3. | Neural Fingerprints..... | 51 |
| 4.2.1.4. | Chemprop Fingerprints..... | 51 |
| 4.3. | Train-Validation (TV) Pairs..... | 52 |
| 4.3.1. | Prediction Distributions..... | 53 |
| 4.3.2. | Repeated Query Compounds..... | 54 |
| 4.3.3. | Performance Compared to Single Prediction..... | 56 |
| 4.3.4. | Query-Only Prediction..... | 58 |
| 4.4. | Validation-Validation (VV) Pairs..... | 61 |
| 4.4.1. | Performance Compared to Single Prediction..... | 61 |
| 4.5. | Training/Validation Split Types..... | 64 |
| 4.5.1. | Scaffold Splitting..... | 64 |
| 4.5.2. | Performance Comparison Between Split Types..... | 65 |
| 4.6. | Pair Types..... | 66 |
| 4.6.1. | Different Pair Type Descriptions..... | 67 |
| 4.6.1.1. | Random Pairs..... | 67 |
| 4.6.1.2. | Matched Molecular Pairs..... | 68 |
| 4.6.1.3. | Similarity Threshold..... | 68 |
| 4.6.2. | Comparison of Pair Types..... | 69 |
| 4.6.3. | Performance Comparison Between Pair Types..... | 71 |
| 4.6.4. | Pair Types with Scaffold Split..... | 74 |
| 4.7. | Pairwise Dataset Masking..... | 75 |
| Chapter 5. | Conclusion..... | 81 |
| References | | 85 |

List of tables

| | | |
|-----|---|----|
| 2.1 | Examined datasets, their sizes and their sources..... | 30 |
| 4.1 | Number of pairs of each validation type under the different pair types and validation splits, for the AurA dataset with 0.2/0.8 valid compound/train compound split. | 74 |

List of figures

| | | |
|------|---|----|
| 1.1 | Moderate correlation between the aqueous concentration required for narcosis in tadpoles and the solubility in olive oil. Also a noted correlation in the length of the aliphatic chain or the alcohols, as reported in [61]. | 2 |
| 1.2 | A Structure-Activity Relationship table from [33]. The \mathbf{R}^1 group on the parent structure (top) is replaced with the substituent in the corresponding column, and the resulting biological activity is listed in the \mathbf{IC}_{50} column. Lower values indicate a stronger activity, as discussed in section 1.1.2. | 3 |
| 1.3 | Various Hammett constant σ 's and partition coefficient π 's as reported in [40]. Used for relating the structure of these compounds to their toxicity to mosquito larvae. | 5 |
| 1.4 | A molecule encoded as several different SMILES strings, demonstrating the one-to-many relationship. | 8 |
| 1.5 | A vector of various descriptors used to make a molecular fingerprint. | 9 |
| 1.6 | Visualization of the steps of both the Morgan molecular isomorphism testing algorithm and the ECFP generation algorithm, showing overlapping steps. | 12 |
| 1.7 | Several local atom environments that would be assigned an atom ID by the ECFP algorithm. | 14 |
| 1.8 | Comparison of the matrix multiplication involved in a Neural Network and how it relates to the simplified pyramid cartoon representation. | 16 |
| 1.9 | Layout of the DNN predictor for the Merck Kaggle challenge showing the parts that were obfuscated. A training set of vectors was provided based on unknown descriptors. | 19 |
| 1.10 | Comparison of Convolutional Neural Network and the Graph Convolutional Network, showing the adaptation of convolution to non-pixel grids. | 21 |
| 1.11 | Diethyl ether molecule with symbolic column vector atom representations. | 22 |
| 1.12 | Message aggregation phase for the Oxygen atom in Diethyl ether. | 22 |
| 1.13 | Update phase for the Oxygen atom in Diethyl ether. | 23 |

| | | |
|------|--|----|
| 1.14 | Readout phase for Diethyl ether..... | 23 |
| 1.15 | An example of a Matched Molecular Pair. A small localized change has a direct impart on a property value. The property in this case is solubility. using data from [17]..... | 26 |
| 2.1 | General flow of information for QSAR property predicting using an MLP. Item in red is the true activity, unknown at validation time but known at training time to train the model. | 31 |
| 3.1 | Number of active ECFP bits across the AurA dataset for different values of fingerprint radius r . Increases active bits as the fingerprint grows are fingerprint collisions being resolved. | 34 |
| 3.2 | Distribution of Fingerprint uniqueness across the AurA dataset..... | 35 |
| 3.3 | Whole-Dataset or "Master" Fingerprint Generation..... | 37 |
| 3.4 | Database Fingerprint Trimming Process..... | 38 |
| 3.5 | Individual RC-ECFP generation..... | 39 |
| 3.6 | Randomly sampled pairwise similarities from ECFP and RCFP. RCFP of radius 3 with an m -cutoff of 7 was used to make fingerprints of length 3101. ECFP of the same length were made and 10,000 pairs of compounds were randomly sampled.. | 41 |
| 3.7 | Randomly sampled pairwise ECFP similarities and their activity differences, 10,000 random pairs of compounds. A scatter plot is shown on the left and a 2d histogram on the right. | 42 |
| 3.8 | Randomly sampled pairwise RCFP similarities and their activity differences, 10,000 random pairs of compounds. A scatter plot is shown on the left and a 2d histogram on the right. | 43 |
| 3.9 | Fingerprint size after trimming with progressively larger m cutoff value for different datasets..... | 44 |
| 3.10 | Performance of both RCFP and ECFP in MLP and Linear predictors, for different fingerprint sizes (determined by varying m -cutoff values) on the AurA validation dataset. Higher is better for R^2 and lower is better for MAE. | 45 |
| 4.1 | Basic flow of information from a pair of compounds to a pairwise prediction. The final prediction is either the difference in activity (Δ) or the probability of a higher property value (\mathbf{p}). | 48 |

| | | |
|------|--|----|
| 4.2 | Possible pair types for a dataset split into training and validation compounds.... | 49 |
| 4.3 | Predictor structures for different tasks and validation types. Items in red are known at training time and used to train the model, but are unknown at validation/inference time. Compounds are represented as vectors by fingerprinting. | 50 |
| 4.4 | Averaging training compounds to make a final activity prediction from multiple pairs in the TV validation case..... | 53 |
| 4.5 | The distribution of activity predictions for the query compounds with the most and least accurate predicted activities from the AurA dataset. Black dotted line is the mean of the individual predictions, the red line is the ground truth activity for the compounds..... | 54 |
| 4.6 | Performance in R^2 and MAE resulting from modifying the number of compounds in the training group of each query compound, called its "training repeat" number. For R^2 higher is better and for MAE lower is better..... | 57 |
| 4.7 | Performance comparison between the pairwise TV approach and the single approach. Measured in R^2 for several fingerprint types and datasets..... | 58 |
| 4.8 | Architecture of query-only predictions. Training compounds are replaced with random vectors sampled from a bit-wise Bernoulli distribution according to average bit densities across the dataset..... | 59 |
| 4.9 | Scatter plots showing A) The performance of a trained model on complete Training/Validation pairs and B) The performance of a trained model on query-only pairs with random vectors as training compounds. These results are for the AurA dataset and ECFP fingerprint type, with a trained predictor network architecture optimized by HPO..... | 60 |
| 4.10 | R^2 performance of various pairwise models trained on random pairs from various dataset in the standard "full pair" validation situations and also in the query-only situation..... | 61 |
| 4.11 | Architecture of the single approach with pairwise sampling, which serves as the comparison baseline for the VV validation type. The predictors on the left and right are the same single predictor, which was trained on single properties..... | 62 |
| 4.12 | The performance as measured by R^2 on validation/validation pairs for several fingerprint types and datasets. Compared with the performance obtained from pairwise sampling predicted values from the corresponding single approach..... | 63 |

| | | |
|------|--|----|
| 4.13 | The performance as measured by R^2 on training/validation pairs compared between random and scaffold dataset split for several fingerprint types, on the AurA dataset. | 66 |
| 4.14 | The performance as measured by R^2 on training/validation pairs compared between random and scaffold dataset split for several fingerprint types, on the BACE dataset. | 67 |
| 4.15 | The performance as measured by R^2 on training/validation pairs compared between random and scaffold dataset split for several fingerprint types, on the CEP dataset. | 68 |
| 4.16 | Pairwise similarity distributions of the different pair types for the AurA dataset. A sampled subset is used for random pairs, whereas MMPA and Threshold use all available pairs of the given type. Similarity is Tanimoto distance between ECFP of size 2048 and radius 3. Chosen threshold was 0.6. | 70 |
| 4.17 | Activity difference plotted against pairwise similarity, coloured by point density. Similarity is Tanimoto distance between ECFP of size 2048 and radius 3. Chosen threshold was 0.6. | 71 |
| 4.18 | Distribution of the number of MMPA and Threshold pairs each compound has in the dataset. | 72 |
| 4.19 | The performance as measured by R^2 on training/validation pairs compared between random, MMPA and threshold pairs for several fingerprint types and datasets. | 73 |
| 4.20 | Comparison of prediction scatter plots for datasets masked by how many pairs each compound has. Predictions are from a pairwise TV model trained on random pairs and ECFP fingerprints. | 75 |
| 4.21 | Prediction performance as a function of number of either MMPA pairs or Threshold pairs that each compound has in the AurA dataset. The dataset is masked for each performance value, evaluating performance only on pairs with either at least (top) or exactly (bottom) a certain number of pairs in the training set. | 77 |
| 4.22 | Prediction performance of a single predictor as a function of number of MMPA pairs each compound has in each of the datasets. The datasets are masked for each performance value, evaluating performance only on pairs with either at least (top) or exactly (bottom) a certain number of pairs in the training set. | 78 |

List of Abbreviations

| | |
|-------|---|
| SAR | Structure-Activity Relationship |
| SPR | Structure-Property Relationship |
| PCE | Power Conversion Efficiency |
| QSAR | Quantative Structure-Activity Relationship |
| QSPR | Quantative Structure-Property Relationship |
| LBDD | Ligand Based Drug Design |
| SBDD | Structure Based Drug Design |
| HTS | High Throughput Screening |
| VESPR | Valence Shell Electron Pair Repulsion |
| IUPAC | International Union of Pure and Applied Chemistry |

| | |
|--------|--|
| SMILES | Simplified Molecular-Input Line Entry System |
| NLP | Natural Language Processing |
| FP | FingerPrints |
| MACCS | Molecular ACCess System |
| ECFP | Extended Connectivity Fingerprint |
| ML | Machine Learning |
| AI | Artificial Intelligence |
| ERM | Empirical Risk Minimization |
| DNN | Deep Neural Network |
| MLP | Multilayer Perceptron |
| SVM | Support Vector Machine |
| RF | Random Forest |
| RNN | Recurrent Neural Network |

| | |
|--------|--|
| CNN | Convolutional Neural Network |
| MPNN | Message Passing Neural Network |
| GC(N)N | Graph Convolutional (Neural) Network |
| GRU | Gated Recurrent Unit |
| MMP | Matched Molecular Pair |
| MMPA | Matched Molecular Pair Analysis |
| BRICS | Breaking Retrosynthetically Interesting Chemical Substructures |
| DFT | Density Functional Theory |
| LUMO | Lowest Unoccupied Molecular Orbital |
| MSE | Mean Squared Error |
| MAE | Mean Average Error |
| HPO | Hyper-Parameter Optimization |
| RCFP | Reduced Collision (Extended Connectivity) Fingerprints |

| | |
|-----|-------------------------------------|
| PCA | Principal Component Analysis |
| TV | Training/Validation pairwise type |
| VV | Validation/Validation pairwise type |

Acknowledgements

Lots of thanks to my supervisor, Séb, who has been a huge help with research, academic and professional advice since I started this program. Thanks to my research group, Caroline, Marjolaine, Safia, Maria, Jérémie, Assya and Léonard for always having great ideas and insights. Thanks to my professors and teaching assistants, without the right course instruction some of the techniques that I relied on to do these experiments and complete this work would have been completely inaccessible to me.

Chapter 1

Introduction

1.1. Predictive Cheminformatics

1.1.1. Structure-Activity Relationships

Property prediction from molecular structure is an important task in drug discovery, quantum chemistry and material design. For drug discovery specifically, computationally identifying potent compounds from huge libraries or quickly estimating the potency of hypothesized compounds can cut down on the development cost of therapeutics in both time and resources [14, 36]. The fundamental basis for the prediction of compound properties is the understanding that changes in a molecule's composition and structure result in changes in the properties of that molecule [9]. This is because the composition and structure of a molecule dictate how it interacts with other molecules in its environment, which is the basis of molecular activity and most molecular properties [11]. There are other intrinsic properties of molecules, most relating in some way to electronic structure, which again is dictated by chemical structure [72]. The detection and characterization of these structure activity/property relationships (SAR/SPR) is the backbone of predictive cheminformatics.

The earliest examples of such analysis dates back to the late 19th century with the observation that an increase in the aqueous solubility of aliphatic alcohols results in a lower toxicity of those same alcohols [68]. An investigation into a very similar trend by Meyer and Overton reveals a similar relationship between solubility in olive oil and relative efficacy of anesthetics [61]. Going beyond just a property-property correlation, the authors observe the trend in the aliphatic carbon chain length of the molecules. This correlation between structure and activity is shown in figure 1.1.

Recognizing SAR and SPR remains a key part of drug and material development [96, 1, 45]. From the drug/biological side, whenever there is a target of interest, SAR tables are often constructed to better understand the structural source of activity. Figure 1.2 is an example of such a table and shows that for a small number of similar compounds, these

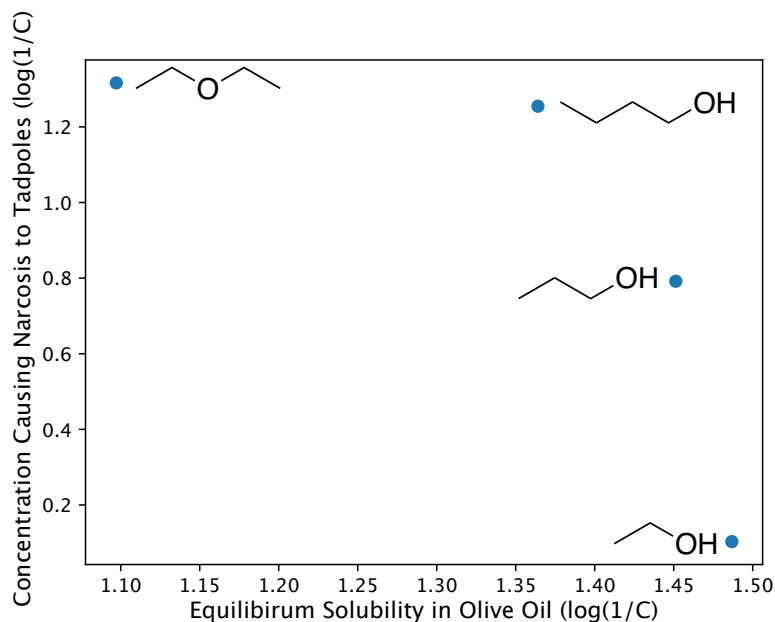


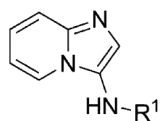
Fig. 1.1. Moderate correlation between the aqueous concentration required for narcosis in tadpoles and the solubility in olive oil. Also a noted correlation in the length of the aliphatic chain or the alcohols, as reported in [61].

tables can be parsed quite easily by human chemists. Chemists can intuitively map out which substructures result in increases or decreases in activity. However, when the number of compounds in a SAR study becomes too large to quickly digest the underlying trends by simple observation, computational methods can be used to automate the process.

1.1.2. Properties and Activities

The above section makes reference to several kinds of molecular properties that can be predicted from chemical structure. Properties of molecules can range from intrinsic features of the compounds themselves, like molecular weight and melting point, to properties of the compound in a bulk medium, like solubility in water, up to even more complex properties that involve the compound's interactions with specific proteins. Properties of compounds can often be intertwined and correlate with each other, because on some level they all find their basis in structure [86]. It is for this reason that the earliest prediction methods for properties were directly based off of other properties, as shown in figure 1.1.

The ability of a molecule to affect the function of a biological entity, such as a protein, cell line or even multi-cellular organism, is called a "biological activity" or a "bioactivity".



| Cpd | $\text{R}^1=$ | IC_{50} (uPA) [μM] |
|-----|---------------|--|
| 14h | | ~250 |
| 15a | | 9.04 ± 0.62 |
| 15b | | 19.39 ± 1.22 |
| 15c | | ~200 |
| 15d | | >1000 |
| 15e | | 500 |
| 15f | | >500 |
| 15g | | 500 |

Fig. 1.2. A Structure-Activity Relationship table from [33]. The R^1 group on the parent structure (top) is replaced with the substituent in the corresponding column, and the resulting biological activity is listed in the IC_{50} column. Lower values indicate a stronger activity, as discussed in section 1.1.2.

The term activity differentiates these properties from the more intrinsic molecular properties like solubility and melting point [65]. In this way, bioactivity prediction is a subset of the larger problem of property prediction.

A compound is "biologically active" for a protein, cell line or organism if it has a negative or positive effect on the function or processes of the protein, cell or organism. Within these activities, so-called "inhibition activity" is most common and it quantifies how effectively a compound inhibits the function of a protein or the growth of a cell line or organism. Inhibition activity is normally expressed in terms of the concentration necessary to inhibit 50% of the function or growth of a given entity. This value is called IC50, and is the target property in figure 1.2 [90]. IC50 is expressed as a concentration which can span several orders of magnitude, depending the compounds, so pIC50 is its practical form [76]. pIC50 is a log-adjusted measure of concentration similar to pH, showed in equation 1.1.1.

$$\text{pIC50} = -\log(\text{IC50}) \quad (1.1.1)$$

$$\text{pIC50} = \log(1) - \log(\text{IC50}) \quad (1.1.2)$$

$$\text{pIC50} = \log\left(\frac{1}{\text{IC50}}\right) \quad (1.1.3)$$

pIC50 is equivalent to the $\log(1/C)$ in figure 1.1 if C is taken to be the concentration of 50% inhibition. Of the three datasets used in this work, two of the three predicted properties are pIC50 activity values. The other is an electronic property related to the use of the compound in an organic solar cell, called power conversion efficiency (PCE).

1.1.3. Quantitative Structure-Activity Relationships

SAR trends were first quantified into mathematical models in 1964 by the concurrent works of Hansch and Fujita [40] and Free and Wilson [28]. These analyses correlate the concentration of desired biological response to many different identifiers of chemical structure in a multi-variate regression analysis. The structure of a compound of interest is decomposed into substituents (sub-structures) which have independent solubility π and electronic σ properties. Figure 1.3 shows several substituents and their parameters. σ is the Hammett constant, which is a measure of the electron-withdrawing ability [39]. π is the partition coefficient, similar to the solubility in oil mentioned above in section 1.1.1 [29]. The σ and π parameters are determined for each substituent by experiment on the substituent alone. Equation 1.1.4 shows the linear model used to predict "concentration of desired response" (C in the equation) from these substructure parameters. The inverse of concentration of desired response is a value analogous to IC50, discussed above in section 1.1.2. The constants k , k' and k'' are fitted to known data and have some biological meaning relating to the free energy of the compounds moving in biological media and interacting with biological constructs like traversing cell membranes.

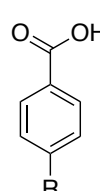








| | R | σ | π | R | σ | π |
|---|--|----------|-------|--|----------|-------|
|  |  F | 0.06 | 0.22 |  CH ₃ | -0.17 | 0.43 |
| |  Cl | 0.37 | 0.81 |  OH | -0.36 | -0.27 |
| |  Br | 0.23 | 1.01 |  NO ₂ | 0.78 | 0.04 |
| |  I | 0.28 | 1.17 |  OCH ₃ | -0.36 | 0.11 |

Fig. 1.3. Various Hammett constant σ 's and partition coefficient π 's as reported in [40]. Used for relating the structure of these compounds to their toxicity to mosquito larvae.

$$\log \frac{1}{C} = -k\pi^2 + k'\pi + \rho\sigma + k'' \quad (1.1.4)$$

These methods are collectively referred to as Quantitative Structure Activity/Property Relationships (QSAR/QSPR) and are a significant portion of cheminformatics. In the years since the development of these early models, computing power has increased by many orders of magnitude and structural chemistry knowledge has matured, vastly improving QSAR/QSPR.

1.1.4. Ligand-based and Structure-Based Drug Design

Bioactivity prediction is a prominent subset of property prediction, and is a main motivation for this work. Therefore, an introduction to bioactivity-specific prediction methods is warranted. There are several approaches to bioactivity prediction, depending on the level of known information about the biological target. The early QSAR methods mentioned above in section 1.1.3 focus on predicting biological properties using compound structure alone, which makes it a Ligand-Based Drug Design (LBDD) approach [3]. If the desired activity involves a single protein, and the structure of that protein is known, then the structural information of the protein is certainly useful to understanding the compound-protein interaction. Using this structural info to aid in understanding/predicting activities is referred to as Structure Based Drug Design (SBDD) [48].

SBDD is obviously more information-rich than LBDD, but carries with it some specific challenges. For example, the placement of the ligand molecule in interaction with the protein. A binding site needs to be identified, then a correct binding pose of the ligand within that binding site. This adds multiple layers of computational complexity which results in the scope of application of SBDD being limited however if structural information regarding a target is known, there is a large advantage to incorporating it [53].

When structural information is not known, it can be due to factors such as difficulties with structure determination. However, SBDD is only applicable in the specific case of single-protein activity prediction, and thus for other prediction tasks, something akin to ligand-based design is used. Non-biological properties like solubility, melting point and PCE are examples of these [38]. This work focuses on developing new QSAR models and therefore describes LBDD, with no included protein information. This is for ease of portability between bioactivity datasets and to other non-biological property targets.

1.1.5. Virtual Screening

Another term that gets mentioned quite often in the area of predictive cheminformatics is that of "Virtual Screening". This is an umbrella term which means any computation method that serves as a surrogate for an in-laboratory High Throughput Screening (HTS) experiment. HTS experiments are large scale screens to evaluate the properties or activities of entire libraries of compounds [4]. These screens are typically part of the early steps in drug discovery, focusing on identifying "hits" (compounds of above-average activity) to be further investigated in other experiments, or to be pushed through to the next steps of the drug discovery pipeline. A further introduction to the various steps of the drug discovery process is described in section 1.4.1. The appeal of computationally replacing an HTS is to save the time and resources spent performing the screen, synthesizing the compounds or purchasing them from a supplier [15].

The predictive models underlying a Virtual Screen can follow different approaches, depending on the level of accuracy desired, the knowledge of the target and the compound dataset that is being screened. "QSAR virtual screens", for example, use QSAR methods like the ones mentioned above. They take a training set of known compounds along with a set of new compounds to make predictions about possible hits [63]. SBDD or "docking" virtual screens rely on known protein structures and make decisions on which compounds are hits by using scoring functions or simulated docking of ligands and protein structures [42]. For these methods, a training set is not required, but the protein structure is required, as well as a general knowledge of how the protein behaves in a biological context. Because the goal of virtual screening is to approximate HTS experiments, it is a broad class of methods. The approaches described in this thesis do not align very well with virtual screening as a field, because we are concerned with exact activity prediction over hit/not-hit identification.

1.2. Chemical Representations

1.2.1. Making Good Chemical Representations

The natural form of chemical compounds in a biological setting (dissolved in aqueous solution) are ensembles of many different interchanging three dimensional conformers. Their representations can be simplified beyond this to the most common representation for human interpretation, the two dimensional molecular graph. The chemical graph encodes all the necessary connectivity information between the atoms. Three dimensional conformer shapes can be derived from these graphs using a chemist’s geometric intuition and tools like Valence Shell Electron Pair Repulsion (VESPR) theory [31]. Although three dimensional shapes may be a more correct idea of compounds in nature, they can be transient forms and the molecule may exchange between several of them. Atomic connectivity, however, is unchangeable without altering the identity of the molecule through a chemical reaction, so it remains the paramount chemical representation and therefore graphs are the way compounds are represented in this work.

Interpreting graphs is a difficult problem in computer science. Variable size and connectivity make the space of possible graphs quite large and difficult to quantify. The computational complexity of important graph tasks varies wildly. For example, searching the nodes in a graph by traversing it is NP-complete [50], confirming the equivalency of two identical but differently labeled graphs is of unknown complexity but is likely NP-intermediate [62], and finally searching for subgraphs within a graph is NP-hard [98]. These are all tasks that are useful in molecular manipulation and chemical informatics and are made difficult by the nature of the data.

The International Union of Pure and Applied Chemistry (IUPAC) has listed several desired traits for good chemical representations, such a uniqueness and ease of manipulation with computer systems [19]. For a chemical representation to be suited to a predictive task, there are also desired traits, some of which overlap with the IUPAC list. For example - it is preferable if molecules are represented outside of any coordinate system and instead focuses on the atomic connectivity alone. This makes the representation of the molecule invariant to translations, rotations and node-relabelling. Also, chemical similarity should be preserved in the representation. This means that chemically similar molecules with similar structures, compositions and sizes should exhibit a similar level of similarity in the representation. This would mean that the representation captures the scope of chemical space well. Another facet of a good representation would be that the representation of a molecule contains the

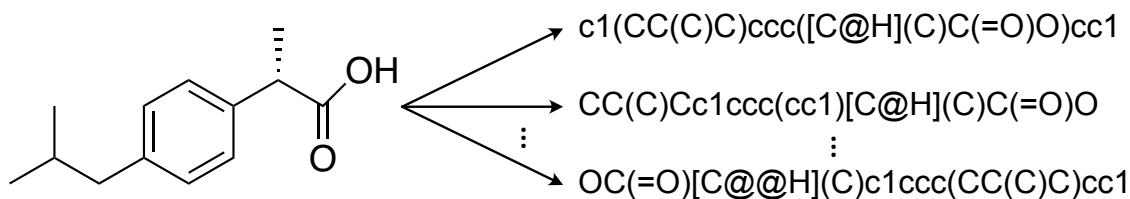


Fig. 1.4. A molecule encoded as several different SMILES strings, demonstrating the one-to-many relationship.

representations of portions of that molecule. Graphs obey this property through their substructures, but other representations do not always obey this property, such as canonical SMILES strings which are discussed below.

1.2.2. SMILES

Simplified Molecular-Input Line Entry System (SMILES) is a relatively simple grammar for representing compounds in a human-readable single line format, and for the input of molecules using a keyboard [91]. Special characters and symbols represent chemically relevant structures and configurations such as different bonds, molecular rings, branches and stereoisomers. SMILES is a general set of rules, therefore many valid SMILES strings exist for a given compound. Figure 1.4 shows several SMILES strings that are valid for an example compound.

This one-to-many relationship makes using SMILES strings a challenge for predictive cheminformatics. In addition, SMILES string length varying with the size of the molecule can cause problems as some predictive models struggle with varying input size. The many-to-one relationship is made a one-to-one relationship through a canonical generation process that makes the general set of rules much stricter to produce a single string for each molecule. Possibly the largest drawback of SMILES is that it is a representation that does not seek to preserve similarity, so physical similarity/structural overlap in the compound space is not preserved in their canonical SMILES strings [47]. However, SMILES has seen some recent successes in the predictive space, as well as successes in molecular generation, mainly using machine learning methods developed from Natural Language Processing (NLP) and translation methods [37, 74]. In this work however, SMILES strings are only used as a means to store the chemical structures in files before converting them to a more powerful and expressive representation for prediction.

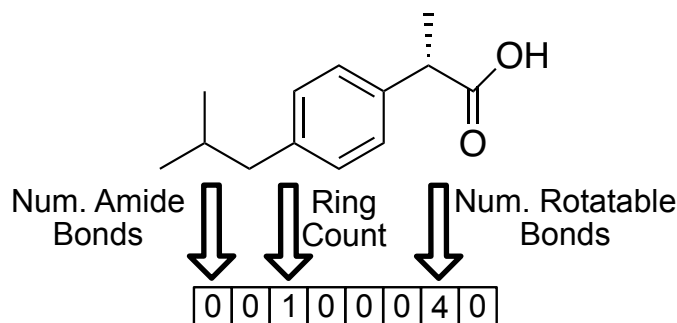


Fig. 1.5. A vector of various descriptors used to make a molecular fingerprint.

1.2.3. Chemical Descriptors

In section 1.1.3, compounds were represented as a collection of specific sub-structures with specific π and σ parameter values. Although this representation is physically meaningful (those parameters have a physical meaning regarding the electronic or solubility properties of those substructures), its representation power is limited. Primarily, the substructure space is large and substructures cannot possibly all be enumerated and tested for π and σ values, like the ones in figure 1.3. Secondly, the π and σ parameter approach relies on linear additivity of the parameters for the various substructures in a molecule, which is not always guaranteed especially for overlapping substructures.

The structure-parameter based chemical representations are an example of "Chemical Descriptors", which are small units of chemical information you can assign to molecules, so as to quantify their structure. Mauri et al. 2017 lists several requirements for molecular descriptors, which match the above-mentioned requirements for molecular representations. First, descriptors must be invariant to atom relabelling and numbering. They must be invariant to molecular roto-translation. They must be defined by an unambiguous algorithm and finally they must have a well-defined applicability on molecular structures [60]. Examples of single-value chemical descriptors include solubility, number of rings, solvent accessible surface area, number of rotatable bonds, number of hydrogen bond donors, and many others. They are sometimes referred to as "zero dimensional" chemical descriptors to contrast against fingerprint vectors, which are called "one dimensional" descriptors. Tensor rank would be a more appropriate term to describe their difference instead of dimensionality, but the term dimension is commonly used. Figure 1.5 shows several of these descriptors.

Grouping several of these single-value descriptors produces a crude version of a molecular fingerprint. A fingerprint, or a one dimensional chemical descriptor, is an information-dense

and explainable representations of an entire compound. A one dimensional vector of the same descriptors in the same order for two different molecules will provide a representation that is comparable between them. However, there are not enough simple single-value structural descriptors to fully capture the massive space of different chemical compounds, so better representations are needed for most tasks.

1.2.4. Fingerprints

Fixed-length vectorial representations of entire molecules, called fingerprints (FP's) and sometimes one dimensional descriptors [60], were developed to solve several of the problems in chemical representations. Primarily, for predictive systems, as well as for tasks such as quick database similarity searching, a fixed-length fingerprint representation is desired [13]. Secondly, by involving the entire molecule, fingerprints can capture information about the whole structure, not just a handful of select features, like what is available through single value descriptors.

Molecular ACCess System (MACCS) keys [25], developed by MDL Information Systems, are a fixed binary vector of either length 166 or 960 with each bit corresponding to whether a pre-defined substructure is present or absent in a given compound structure. For example, the presence of an amino group, "-NH₂", on a molecule causes bit number 84 to be turned on.

Atom Pair fingerprints [12] aggregate the identities and distances of each pair of atoms within the molecule. For example the structural feature "-CH₂-CH₂-", two adjacent methylene carbons, form a pair of atoms and would be encoded as "CX2-(2)-CX2". These encodings are then counted for different molecules and the counts are used directly in a similarity metric, or are hashed into a fixed length vector.

1.2.5. Extended Connectivity Fingerprints (ECFP)

Extended Connectivity Fingerprints (ECFP) [69] are the most popular fixed-length vectorial representation of molecular graphs and serve as a baseline molecular representation for the predictions in this work. They are based on the Morgan algorithm and for this reason are sometimes called "Morgan Fingerprints". ECFP was created by Accelrys inc. for their "Pipeline Pilot" product [41], but the generation process was published. However, unknown algorithmic specifics (such as hash functions) make an exact re-implementation of Accelrys's version of ECFP technically impossible. So generally, the Pipeline Pilot version of the fingerprints are called ECFP and any open source re-implementations are referred to

as "Morgan Fingerprints". That being said, the names are essentially interchangeable and the term ECFP is not a trademark so it is used in this work to refer to ECFP/Morgan Fingerprints.

As mentioned above, ECFP/Morgan fingerprints are based on (but not equivalent to) the Morgan algorithm. This algorithm was developed to solve the graph isomorphism problem [62]. This task is to determine if two differently labeled graphs are the same, which is important for cheminformatics. The Morgan algorithm is described in algorithm 1.2.1, where the "special_recoding_function" updates atom IDs in a strictly increasing way based on the ID's of an atom's neighbours to prevent overlapping labels. The Morgan algorithm is also shown graphically along with the ECFP algorithm in figure 1.6. It first assigns a descriptor or "ID" to each atom according to its element, valence, number of hydrogens and other features. Then, the features of its neighbouring atoms are incorporated into its own ID in a pre-defined order. This neighbour aggregation process is repeated until each atom has a unique identifier (or as unique as symmetry will allow). The identifiers can then be used as node labels to compare two graphs. Since the labels were generated solely based on node identities and connectivity, no prior node labelling is needed to determine the isomorphism of two differently labelled graphs.

Algorithm 1.2.1. Morgan Canonical Atom Labels

```

Input: Molecule m
  For atom a in m
     $x_a \leftarrow atom\_ID\_lookup(a)$ 
  end For
  While  $x_1 \dots x_n$  are not unique
    For atom a in m
       $x_1 \dots x_n \leftarrow neighbours(a)$ 
       $x_a \leftarrow special\_recoding\_function(x_a, x_1 \dots x_n)$ 
    end For
  end While
  Return  $x_1 \dots x_n$       % For comparing to other molecules

```

The ECFP generation process is similar and described in algorithm 1.2.2 and shown graphically in figure 1.6 alongside the Morgan algorithm. The identifiers for each atom (node) are assigned, and the neighbour aggregation step passes and incorporates the value of these identifiers between neighbours. This aggregation/incorporation step happens a fixed

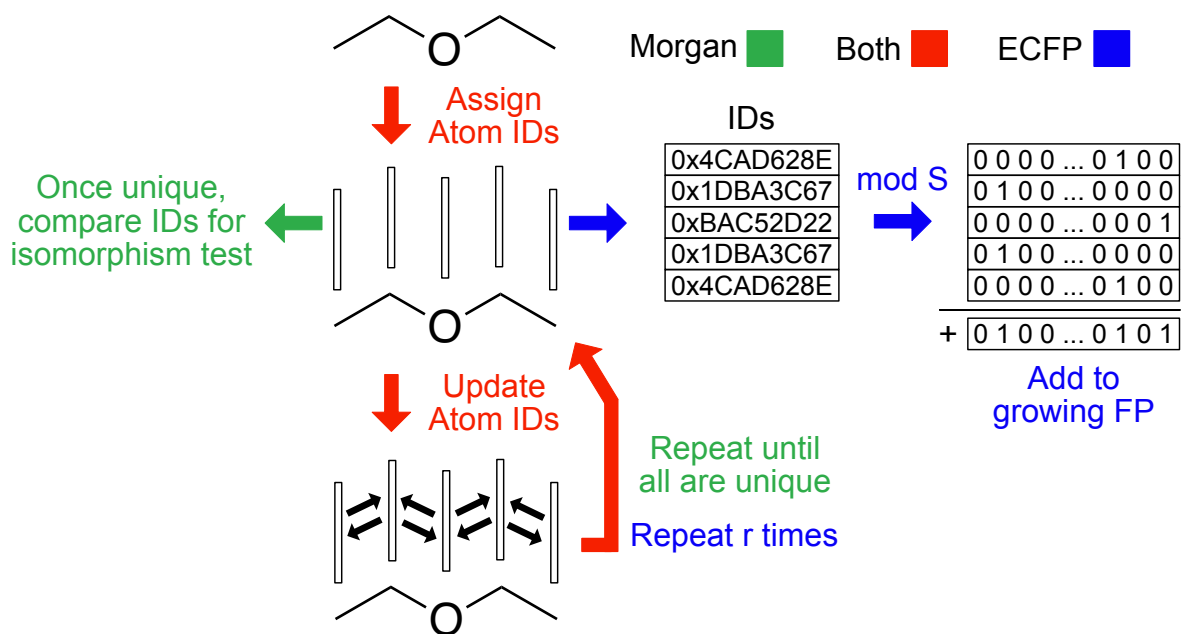


Fig. 1.6. Visualization of the steps of both the Morgan molecular isomorphism testing algorithm and the ECFP generation algorithm, showing overlapping steps.

number of times, which is referred to as the number of fingerprint layers, or the fingerprint radius r . It is called the radius because it corresponds to half of the maximum substructure bond width that the fingerprint can represent. During the aggregation step, each atom identifier incorporates information from 1 more bond length away. After each iteration, the current identifiers for each atom are hashed to a 32-bit integer space and then folded down to a more usable size, usually 2048 bits, corresponding to an 11-bit integer space.

Algorithm 1.2.2. ECFP Generation

```
Input: Molecule  $m$ , radius  $R$ , fingerprint_size  $S$   
For atom  $a$  in  $m$   
     $x_a \leftarrow atom\_features(a)$   
end For  
 $f \leftarrow \mathbf{0}_S$   
For layer  $l$  in  $R$   
    For atom  $a$  in  $m$   
         $x_1 \dots x_n \leftarrow neighbours(a)$   
         $\mathbf{v} \leftarrow [x_a, x_1 \dots x_n]$     % Concatenation  
         $x_a \leftarrow hash(\mathbf{v})$   
         $i \leftarrow mod(x_a, S)$   
         $f_i \leftarrow 1$   
    end For  
end For  
Return  $f$ 
```

When ECFP iterates for multiple fingerprint layers, atomic representation become substructure representations for the local neighbourhood around the central atoms. Figure 1.7 shows several of these local atom environments for two similar molecules. Where the molecules overlap, they share multiple atom environments and where they differ, they have vastly different atom environments. In ECFP, each of these atom environments would be represented by a hashed atom ID for the centre atom, and they would all be added to the fingerprint. Although some atom environments may share some overlapping structure, unless they are identical, they have completely different IDs and therefore are treated completely different.

One of the main advantages is that ECFP takes a graph of any size and any connectivity and produces a fixed-length feature vector for that graph. It does this based on substructures, which have been shown throughout cheminformatics as being a significant contributor to overall compound activity. One of the main disadvantages is that the hashing and folding process can produce hash collisions, where multiple different substructures end up activating the same bit in the final fingerprint. These collisions are expected to be random across the different possible substructures and therefore colliding bits are not expected to be chemically similar at all, which may be confusing to predictive models.

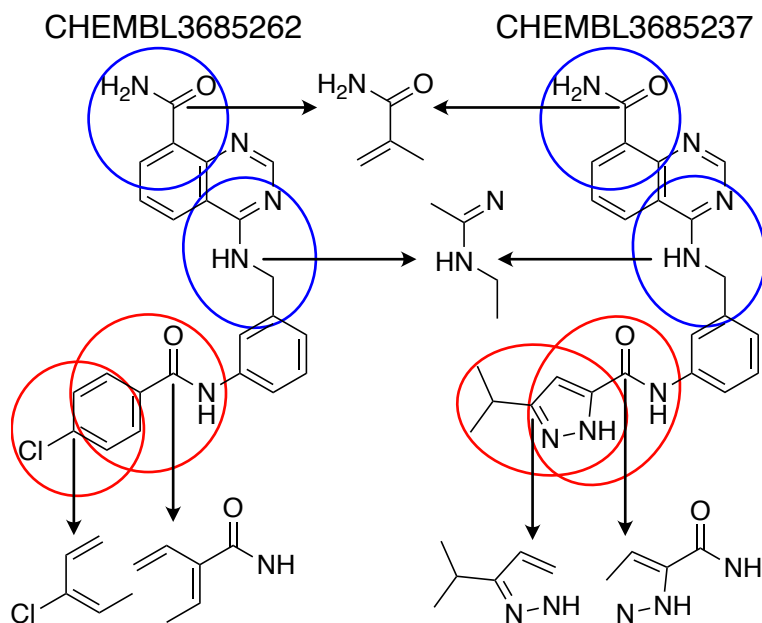


Fig. 1.7. Several local atom environments that would be assigned an atom ID by the ECFP algorithm.

1.3. Machine Learning in Chemistry

1.3.1. Machine Learning (ML)

Machine Learning (ML) is a subset of the field of Artificial Intelligence (AI) that focuses on understanding and gaining insights about data in an automatic way. Building on a collection of known data, ML systems can be "trained" to recognize complex relationships between data points and their corresponding labels. Trained models can then be used to make predictions about data points with unknown labels. This training/testing paradigm is also sometimes referred to as discriminative machine learning [64].

Consider a dataset D , which is composed of paired input-output datapoints (x_i, y_i) where x_i are features and y_i are their corresponding labels. The points of data x_i can be any kind of observations for a prediction task, like the molecules in QSAR. The labels y_i are the properties of the molecules that we are predicting and are either real values for a regression task, or categorical ones for a classification task. In the prediction scenario, we have some data points x_i for which we know the labels y_i , called the training set, and some more data points for which we withhold the labels to simulate datapoints with unknown labels, called the test set. For our purposes, all the datapoints come from the same dataset source so we

divide the dataset into a training set and a validation set, where we withhold the labels of the validation datapoints to evaluate the performance of the trained model. This dataset splitting strategy is discussed further in section 2.1.

The goal of the learning process is to produce a function f that can make accurate predictions of y_i for any given x_i . To do this, a family of functions F is selected that is parameterized by θ , meaning we have access to many functions $f_\theta \in F$ that differ by their parameters θ . F is chosen so that the parameters θ can be easily optimized to achieve good predictions on the training set. The optimized f_θ is then used to make predictions for the x_i values in the validation/test set.

To optimize the predictor f_θ , its performance is first measured using $L(f_\theta(x_i), y_i)$, where L is a function called the loss. The loss is a measurement of the correctness of the predictions, similar to accuracy, but preferably a smooth function. The loss is averaged over all points in the training dataset D . This value is called the Empirical Risk and is shown in equation 1.3.1. This general framework describing the learning process is called Empirical Risk Minimization (ERM) [85].

$$\hat{R}(f, D) = \frac{1}{|D|} \sum_{x,y \in D} L(f(x), y) \quad (1.3.1)$$

The most common way to optimize θ is to evaluate $\frac{\partial \hat{R}}{\partial \theta}$ and use it to adjust θ to reduce the loss [35]. This requires organizing the prediction models so that for every parameter $p \in \theta$, the partial derivative $\frac{\partial L}{\partial p}$ is defined and accessible. The chain rule is applied to break the derivative up and dynamic programming [5] is employed to re-use the partial solutions for an efficiency gain. Applying these two efficiency-boosting approaches to access $\frac{\partial L}{\partial \theta}$ is the backbone of modern neural network training, a process called automatic differentiation or backpropagation [70].

All of the fundamental prediction models used in this work are Feed-Forward Neural Networks, also called Deep Feed-Forward Neural Networks, Deep Neural Networks (DNN) or Multi-Layered Perceptrons (MLP) [35]. Put simply, MLP's are functions that non-linearly transform fixed-length vectors into different vector shapes. For this work, the output of the MLP's are property predictions and are therefore always a single scalar value.

The structure of an MLP is a series of linear transformations interspersed with element-wise non-linearities. The prediction process, or "forward pass", involves iteratively updating a hidden representation \mathbf{h} that begins as the input \mathbf{x} as shown in equation 1.3.2. This hidden representation is updated L times, which is the number of layers specified in the neural network. The length of \mathbf{h} may or may not change size as its being updated. A single

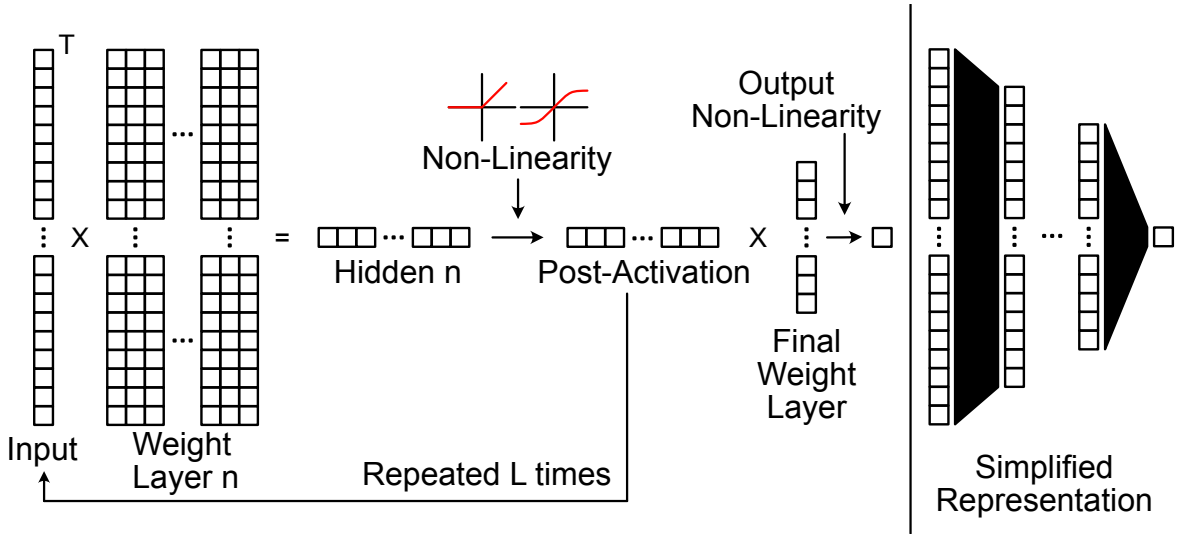


Fig. 1.8. Comparison of the matrix multiplication involved in a Neural Network and how it relates to the simplified pyramid cartoon representation.

one of these updates is shown in equation 1.3.3. τ is the element-wise non-linearity and can be functions such as $\tanh()$ or $\text{ReLU}()/\max(x,0)$. Finally, the output of the model is shown in equation 1.3.4. An output non-linearity \mathbf{o} is applied which can be functions such as $\text{Sigmoid}()$ or $\text{Softmax}()$ for probabilistic/categorical outputs or $\text{Identity}()$ for regression outputs.

$$\mathbf{h}^{(0)} = \mathbf{x} \tag{1.3.2}$$

$$\mathbf{h}^{(k)} = \tau \left(\mathbf{b}^{(k)} + \mathbf{W}^{(k)} \mathbf{h}^{(k-1)} \right) \tag{1.3.3}$$

$$\mathbf{f}(\mathbf{x}) = \mathbf{h}^{(L+1)} = \mathbf{o} \left(\mathbf{b}^{(L+1)} + \mathbf{W}^{(L+1)} \mathbf{h}^{(L)} \right) \tag{1.3.4}$$

A graphical representation of the steps described in equations 1.3.2 to 1.3.4 is shown on the left side of figure 1.8. The right side of the same figure shows the typical simplified representation of an MLP that distills its function down to a black box that produces single scalar predictions from fixed-length input vectors. This pyramid-shaped cartoon is used in this work and others as shorthand for the predictor.

1.3.2. Early ML in Chemistry

QSAR models, such as the basic ones described above in section 1.1.3, fit well into the paradigm of discriminative machine learning. There is a set of known training data and the task for the model is to use insights discovered in the training set to make predictions about the validation/test set. The first QSAR prediction models mentioned in section 1.1.3 were linear regression models which, while not often considered a machine learning method due to simplicity, is a parameterized prediction method that is trained on known data and makes predictions on unknown data, like most ML models.

As mentioned in section 1.2.3, the nonadditivity of descriptors means the problem of QSAR quickly outpaced the capabilities of linear regression models such as σ - π analysis [40]. The relationship between structure and activity is intrinsically non-linear because the structure space is so complex, so the shift to non-linear models was a natural one that resulted in higher performance [34].

Very simple MLPs were first applied to QSAR modelling tasks [2, 22], which were suitable for analyses with dozens of compounds and only a few descriptor features due to the low computational power available at the time. An explosion in possible descriptors and fingerprints meant that the total number of features in typical QSAR investigations quickly became insurmountable for the neural networks of the time without dimensionality reduction or feature pre-selection [79]. This opened the field to other prediction methods for more than a decade from the late 1990's to the early 2010's before the re-emergence of neural networks to prominence in QSAR with the development of deep learning.

During this gap in the prominence of neural networks, Support Vector Machines (SVM) [16] were a prevalent QSAR method and were considered to be among the state-of-the-art [10]. SVMs classify compounds by deciding a max margin hyperplane in the descriptor space to separate compounds of different labels. The label of a datapoint is predicted based on what side of the hyperplane it lies on. In addition to this, SVMs employ what is called the "Kernel Trick" [82], where the distance between the datapoints and the hyperplane are measured using a non-linear distance measurement or "Kernel", which allows for a non-linear decision surface. Random Forests (RF) [8] also became a prevalent method for QSAR prediction around this time. RF's are ensemble models of decisions trees that make several decisions regarding the input feature vector so as to classify or regress on its label values. Their high performance, ease of training, few adjustable parameters and speed of training made them a easy choice for many QSAR applications [79, 59].

All of these predictor types rely on fixed-length vectorial representations of arbitrary chemical graphs, further underlining the importance of good chemical representations for machine learning QSAR prediction models.

1.3.3. Early Deep Learning in Chemistry

Deep Learning is a recent direction of the field of ML that focuses on models with an extremely high number of parameters. The term normally applies to MLP’s with many hidden layers, making them deep, and usually many hidden nodes per layer, making them wide as well. They are often referred to as DNNs but they are fundamentally the same predictor. DNNs joined the mainstream of QSAR prediction in the wake of the Merck bioactivity Kaggle challenge from 2012 [59].

As an open-source contest, contestants attempted to train a QSAR prediction model with the highest overall accuracy on a hidden dataset. The contest participants were given a training dataset consisting of descriptor vectors representing compounds. These descriptor vectors varied in length across the different targets of the competition from 4306 descriptors up to 12508. The compound structures themselves remained proprietary to Merck, as did the exact descriptors or fingerprints that were used, which makes for a very uninterpretable predictor.

The winning team used DNNs, ensembled with some other machine learning approaches, but it was suspected that DNNs were what was imparting the majority of the prediction power [59]. The DNN prediction framework for this problem is shown in figure 1.9, showing the obfuscated nature of the predictor. Merck as an organization remained intrigued by the positive results and the ability of DNNs to out-perform their most advanced internal methods. The winning submission was followed with efforts from the same group at Merck to demystify the black-box nature of the neural network predictions [95].

In the aftermath of this competition, there was a new renaissance for neural networks in predictive chemoinformatic models. The multi-task deep neural network models developed during the competition outperformed some of the industry standard commercial predictive software. This attracted the large pharmaceutical companies towards the new field of Deep Learning [83]. DNNs operating on fixed-length descriptors/fingerprints remained the prominent predictive cheminformatic models until the development of Message Passing Neural Networks, discussed below in section 1.3.5.

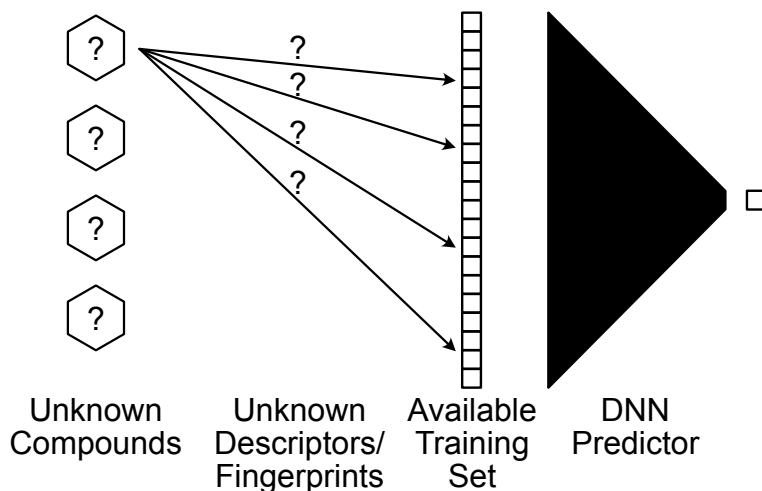


Fig. 1.9. Layout of the DNN predictor for the Merck Kaggle challenge showing the parts that were obfuscated. A training set of vectors was provided based on unknown descriptors.

1.3.4. Deep Learning in Other Fields

MLP's like the ones described in section 1.3.1 treat all inputs as being of equal importance and have no prior assumptions over the relation between any subset of input variables relative to each other. This lack of special consideration is appropriate for chemical descriptors and fingerprints where hash functions ensure that substructures are represented essentially randomly and uncorrelated across the input vector positions. However, few areas of application offer raw data of this type, so specialized neural networks have been developed that cater to specific input data scenarios.

In the the field of NLP and machine translation, the raw data is strings of words which are members of a vocabulary of possible words, arranged in very specific orders [99]. For the most part, adjacent words tend to depend on each other much more than distant words. So, for data with this kind of temporal structure, the Recurrent Neural Network (RNN) was developed [71]. RNN's incorporate a hidden state containing information on the series of data up to that point. This way, additional information for each time step or word is incorporated into the hidden representation of the input as a whole.

In the field of computer vision and image processing, the raw data is a two dimensional grid of pixels, sometimes with multiple colour channels [89]. In an image, adjacent pixels are much more correlated to each other than pixels from opposite corners of the image. This, plus the very large dimensionality of the input, has led to the development of the Convolutional Neural Network (CNN) [57]. By employing "convolutional kernels", which are two

dimensional weight matrices that are translated across the image and trained using gradient methods, information about the neighbourhood surrounding a pixel can be recognized, such as edges, curves and other elementary features in a translation-invariant way.

Predictive cheminformatics also has a more complicated kind of raw data, the chemical graph. As described in section 1.2, graphs can be algorithmically transformed into fixed length vectors (fingerprints) with uncorrelated bit positions, but the translation to this form is still a loss of information when coming from the 2D graph. To better represent graphs for machine interpretation, the convolution operation from CNNs was expanded to operate on arbitrary graph structures instead of the pixel grids of images [26, 49]. This advancement has applications in all fields with graph data, such as social networks, financial transaction records and of course, chemistry and chemical graphs [101].

1.3.5. Message Passing Neural Networks (MPNNs) in Chemistry

The next truly paradigm-altering development in the field of predictive cheminformatics were attempts to learn directly from the chemical graphs, instead of through fingerprints or descriptor vectors which are a lossy approximation of the real graph data. Learning directly from the molecules allows the learning system to access more structural information that may be missed by the fingerprinting process. The generalization of the convolution operation to generic graphs was a major contribution towards creating a fully learnable fingerprinting process.

When a "same"-type convolution operation (the size of the input and output are the same) is applied to an image, the operation can be viewed as "updating" the pixel values in the image with the results of the convolution. A convolution is essentially a two dimensional dot product between a weight matrix (filter) and an equivalently sized patch of input pixels. The convolution results will "update" that pixel which now contains information not only from itself but also from its neighbours. The Graph Convolutional (Neural) Network approach (GCN/GCNN) see this operation generalized to arbitrary graph structures [94]. Figure 1.10 shows the analogous grid and graph approaches as they may be applied to image grids and molecules.

One major difference between the two approaches is that in a graph there is no concept of relative direction between a node and its neighbours. Without this, a two dimensional weight filter is not viable as the position of a node's neighbours is meaningless and the connectivity alone is what matters. To mitigate this, the convolution is conducted with a common weight for all the neighbours of a node, sometimes adjusted based on edge information.

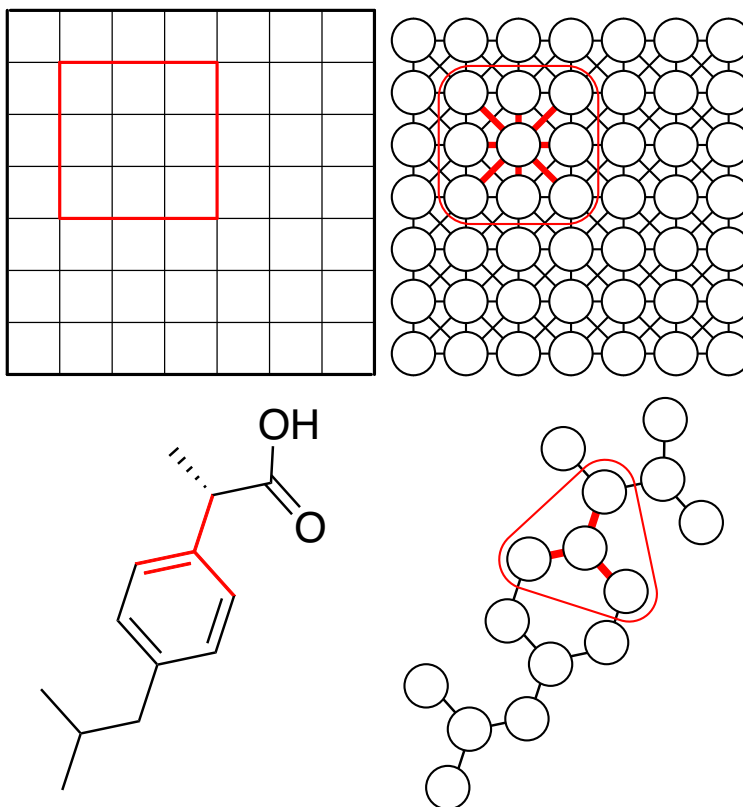


Fig. 1.10. Comparison of Convolutional Neural Network and the Graph Convolutional Network, showing the adaptation of convolution to non-pixel grids.

The GCN approach was first applied to chemical graphs in Duvenaud et al. [26] and Kearnes et al. [49]. The so-called "neural" or "learned" fingerprints are very similar in concept to the ECFP approach, with node receiving information from neighbouring nodes. The learnable weight matrices, however, mean that the fingerprinting process has the ability to learn which substructures are important for a given prediction task. This contrasts with ECFP, which algorithmically represents every substructure (up to a certain radius) in the final fingerprint regardless of its importance for the task. A subsequent work by Gilmer et al. [32] proposes the Message Passing Neural Network (MPNN) framework, which consolidates the theory behind the flow of information in many different graph convolutional networks into three ubiquitous processing steps; the message step, the update step and the readout step.

In preparation for MPNN processing, node representations are generated for each atom in the molecule. Their representations are based on the node's identity and immediate connectivity, but not on its position or label in the molecule. Figure 1.11 symbolizes these

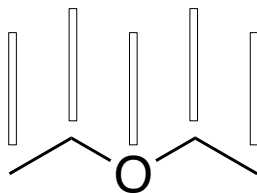


Fig. 1.11. Diethyl ether molecule with symbolic column vector atom representations.

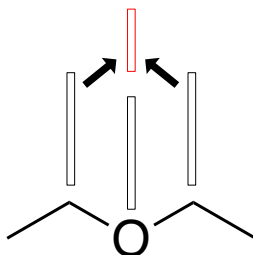


Fig. 1.12. Message aggregation phase for the Oxygen atom in Diethyl ether.

node representations as column vectors for each of the atoms in a Diethyl ether molecule, which will be used to describe each of the steps.

The message function gathers the vectors of all of a node's neighbours and combines them into a single message vector. This is shown graphically in figure 1.12 by focusing on one node, the "target node". In practice this operation is applied simultaneously for all nodes in the graph. The message function uses the source node representation, the target node representation and the representation of the edge that connects them to make a message. It then aggregates all these messages from the various source nodes to a single target node. A general equation for this step is shown in equation 1.3.5.

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}) \quad (1.3.5)$$

The update function takes a node representation and the aggregated message from the previous step and updates the node's representation. Focusing on a single node again, this is shown graphically in figure 1.13 and a general equation for this step is shown in 1.3.6. Some success has been seen in using methods developed for NLP as the update function, such as the Gated Recurrent Unit (GRU), a type of RNN [58].

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \quad (1.3.6)$$

The readout function aggregates the node representations to compute a representation for the whole graph. Figure 1.14 shows a general graphical representation of this. Because

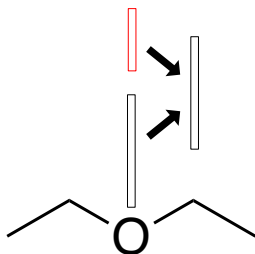


Fig. 1.13. Update phase for the Oxygen atom in Diethyl ether.

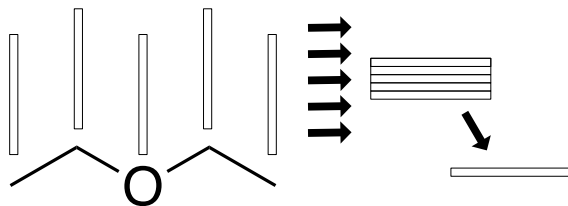


Fig. 1.14. Readout phase for Diethyl ether.

there is no concept of node ordering in a chemical graph, the aggregation process must be completely invariant to ordering. A general equation for this is shown in 1.3.7. Examples of suitable aggregation functions include sum and attention functions [87].

$$\hat{y} = R(\{h_v^T | v \in G\}) \quad (1.3.7)$$

This general MPNN framework of message, update and readout phases is used to introduce the two GCN fingerprinting approaches used in this work. The first of these is one of the first GCN approaches developed for chemistry, Neural Fingerprints as described in Duvenaud et al. 2015. The second is a more recent GCN approach, Chemprop Fingerprints from Yang et al. 2019. The specific message, update and readout phases of both of these approaches are discussed later in section 4.2.

1.4. Medicinal Chemistry Intuition

This work attempts to maintain a general position when it comes to the molecular property being predicted, for ease of application to many different kinds of properties. However, bioactivity targets remain a core motivation for this work. One of the central contributions of this work, the pairwise comparison approach, section 4, relies on Matched Molecule Pair Analysis (MMPA) which is a design tool developed for the drug development process. The drug development pipeline is introduced in this section to give a complete introduction to the field before introducing MMPA. As such, this section deviates from general property

prediction to mostly bioactivity prediction. In addition, two of the three datasets used in this work, described in 2.1, use bioactivity targets.

1.4.1. The Drug Development Pipeline

The drug development process seeks to create new therapeutics that are both safe and effective at treating diseases. The drug development "Pipeline" is a sequential order of developments steps leading to a new drug for a particular disease, and it is generally divided into four specific phases [100].

1.4.1.1. The Discovery Phase. Consists of selection of the disease and identification of the target, among other things. Disease selection is decided both on scientific and commercial grounds, which can be based on things like the severity of the disease, the number of people currently afflicted, and the number projected to be afflicted. Upon selecting a disease, target identification is a more extensive process that may require biological experiments, depending on how well the chosen disease is understood. With the disease and target selected, the search for active compounds for the target can begin. As mentioned briefly in section 1.1.5, a "hit" compound is one found to have above average activity for a target and serves as a basis for further investigation. Libraries of compounds are often screened in the discovery phase to identify hits [52].

1.4.1.2. The Preclinical Development Phase. Begins with the discovered hit compounds, which are then optimized to improve efficacy and pharmacokinetics. Pharmacokinetics are properties of a potential drug that include the ADME properties; absorption, distribution, metabolism, and excretion. Referred to as the hit-to-lead and lead development processes [52, 20], these investigations ensure a new drug must be safe, easily absorbed into the body, present in the body long enough without being metabolized or excreted, and effective against its target. Medicinal chemists optimize for these various properties to produce a molecule suitable for application as a clinical candidate. Also in this phase, processes for industrial synthesis and administration formulations are specified.

1.4.1.3. The Clinical Development Phase. The clinical development phase is divided into 3 phases of clinical trials to evaluate safety and efficacy. The first sets safe maximum dosage levels on a small group of healthy participants. The second tests for efficacy in a small group to help plan the third. Finally, the third phase is a large scale efficacy test conducted over a long period of time.

1.4.1.4. The Product Approval and Launch Phase. The drug candidate is evaluated by regulatory agencies for approval. Upon successful approval, the drug is marketed and additional clinical trials may be conducted for long-term side effects and alterations to the approved specifics of the drug.

1.4.2. Computational Methods In The Pipeline

Although every step could likely be augmented with computational approaches, this work focuses on the medicinal chemistry portion of the development process. This portion is divided into several tasks; hit identification, hit-to-lead development and lead optimization. These tasks are at the interface of the discovery and preclinical phases of the pipeline [100]. Generally, the first of the tasks identifies a compound or class of compounds with some activity towards a desired target. These hits can be selected from libraries or repositories of compounds with unknown activities. The second and third part part refines the chosen structures (hits) iteratively, progressing towards compounds that will seek approval from regulatory bodies [48].

Identifying hits for a given target is a difficult task owing to the size of the chemical space. The number of possible drug-like compounds is truly massive, with a wide range of estimates but a commonly quoted number is in the neighbourhood of 10^{60} [67]. The advantage of computational methods for this task is to rapidly and cheaply search this extensive space, starting with many thousands of compounds in libraries or commercial catalogues like ZINC [44] and narrowing in on only a few compounds for further development. Virtual Screening, mentioned in section 1.1.5, is the common name for computational methods applied to this task.

In the second and third tasks, hit-to-lead optimization and lead development, the problem becomes an interesting multi-parameter optimization problem with many different individual directions to explore. As a structure is being iteratively changed and optimized synthetically, having reliable estimations of the properties of hypothetical compounds allow for bad structural avenues to be pruned off more quickly. To aid in these tasks, a common tool is Matched Molecular Pair Analysis.

1.4.3. Molecular Matched Pair Analysis

MMPA is a design paradigm to rationalize and better understand the high dimensional optimization problem that is hit-to-lead and lead optimization. The term was first coined by Kenny and Sadowski in 2005 [51]. The idea is to compare compounds that differ in

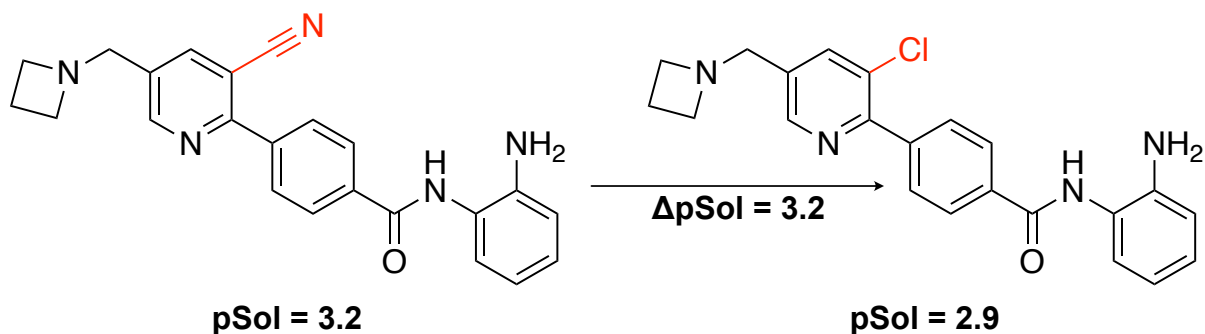


Fig. 1.15. An example of a Matched Molecular Pair. A small localized change has a direct impact on a property value. The property in this case is solubility. using data from [17].

structure only at a specific site, and observing the difference in their properties due to the structural change. Figure 1.15 shows an example of a Matched Molecular Pair, with the target property being solubility. The overlapping structure makes the change in activity more easily understood and explained by a human chemist. MMPA fits the way medicinal chemists may naturally go about explaining a structure activity relationship [23].

In a typical workflow involving MMPA, a database of compounds with known activities is broken down into MMP's. These MMP's are organized into MMP "rules", which are specific structural changes and their resulting property changes. They can be aggregated from the different times the same transformation has occurred across a database and include statistics such as average property change and standard deviation [18]. The ensemble rules can then be stored in a database and referred to for new compound pairs. Traditionally, MMPA actually features no modelling or extrapolation beyond these average property change statistics, it simply consists of a database of rules that are collected and presented in a way to help guide the useful decisions of a chemist [56].

QSAR and machine learning approaches have been applied to MMPA, predicting the difference in activity between two compounds in a matched molecular pair [73]. In that work, a dataset of pairs is carefully selected on the basis of substructure overlap and their membership as part of either the training or validation sets. Pairs are pre-computed using the Breaking Retrosynthetically Interesting Chemical Substructures (BRICS) algorithm [21] before being used to train the model. Compounds are broken down and regarded as being composed of one or more fragments attached to a central core.

As the pairs are being generated, they are also carefully divided into different training and validation sets on the basis of two different validation scenarios. The first involving compound fragments unseen in the training set, the second involving central cores unseen

in the training set. The pairs are represented as three Morgan fingerprints, two for the two different fragments and one for their overlapping central core, all concatenated together. This described ML/MMPA work mainly serves as a comparison between three different ML models on this task; an MLP, a Random Forest, and a Gradient-Boosted Machine, of which it finds the MLP the most performant in the prediction task [73].

1.5. Project Goals and Approaches

The goal of this work is to explore common representations of chemical compounds and to propose more powerful and explainable representations. This goal is achieved through the two different efforts, Reduced Collision Fingerprints and Pairwise Compound Comparisons. Within the scope of the drug development pipeline, we want to develop QSAR tools for a specific phase of medicinal chemistry development. The hit-to-lead optimization and lead development phases, where optimization becomes a complex task balancing multiple properties, and datasets are of modest size.

The rest of this work is structured as follows: The datasets that were used to validate the models, as well as implementation details, and the description of the baseline approaches are found in section 2. Following that, Reduced Collisions Fingerprints are proposed, which are a new molecular fingerprint type to remove the ambiguity resulting from fingerprint hashing collisions. The motivation, methodology and performance of these fingerprints are found in section 3. Finally, the Pairwise Compound Comparison approach is proposed, which is an ML prediction method similar to MMPA, but generalized in certain ways compared to MMPA. The strict pair requirements are relaxed, different pair sources are used, and several kinds of fingerprints are used. The motivations, methodology and implementation of the pairwise comparison approach are described in section 4.

Chapter 2

Datasets and Implementation

2.1. Datasets

Table 2.1 shows properties of the various datasets used for testing and validation of the described methods. Our attempt was to select datasets that have been used to benchmark other property prediction methods and also span different molecular prediction tasks. Across each dataset, identical compounds with differing properties were averaged. For the dataset splits, we used a binary training/validation split throughout this work, with 80% of the data being training and 20% being validation to allow for five-fold cross validation. Although a three-way train/validation/test data split is common through the machine learning literature, in this work a binary split is used for simplicity. We felt that with cross validation and the binary split, a good estimation of performance is obtained without the need for a test set. Perhaps if we had a dataset of particular importance or interest and were researching an underlying phenomenon in the data, the inclusion of a test set with stringently withheld labels would be more appropriate.

2.1.1. AurA

2430 compounds that are inhibitors of the human aurora A (AurA) tyrosine kinase. All activity values were sourced from ChEMBL [30], where the AurA kinase is targetID ChEMBL4722. All compounds with listed activities against this target were collected. These compounds were filtered to leave only those with IC₅₀ values. IC₅₀ values were log transformed to pIC₅₀ values for prediction. This target was also used in Turk et al. 2017 [73], the work introduced in section 1.4.3 with a similar MMPA/ML approach to the pairwise approach described in section 4. This final dataset contains activities coming from different experimental assays, which may introduce experimental noise into the dataset.

| Name | Size | Data Reference | As Seen In |
|------|-------|----------------|------------|
| AurA | 2430 | [30] | [73] |
| BACE | 1513 | [78] | [97] |
| CEP | 29978 | [38] | [26] |

Table 2.1. Examined datasets, their sizes and their sources.

2.1.2. BACE

1513 compounds that are inhibitors of the human β -secretase 1 enzyme. The dataset is available through the MolNet molecular prediction benchmarking package, part of the larger DeepChem cheminformatics package, and therefore has performance as reported by many different methods [93].

2.1.3. CEP

29978 compounds poised as potential substrates for organic photo-voltaic solar cells, curated by the Harvard Clean Energy Project. The target property is PCE, which is calculated from density functional theory (DFT) calculations, using the band gap and lowest unoccupied molecular orbital (LUMO). Although this value is technically a percentage, and is bounded by 0 and 100, no special consideration is taken for this prediction beyond standardizing the database prior to training. Other works predicting this value also make no special considerations for the bounded percentage aspect of the value [26]

2.2. Implementation Details

The methods described in this work were implemented in the Python programming language [84] using the PyTorch deep learning library [66] along with the PyTorch-geometric extension for graph convolutional operations [27]. RDKit [55] was used to manage chemistry and cheminformatics tasks such as SMILES canonization, scaffold fragmentation according to the Bemis Murcko algorithm [6] and generation of matched molecular pairs according to a common fragment-indexing algorithm [43].

The majority of all the models were trained on either a workstation featuring an AMD 32 core 3970X processor, 197 GB of memory and an Nvidia RTX 2080Ti with 11GB of video memory, or on a server featuring 2 Intel Xeon Gold 6130 16 core processors, 395 GB of memory and four Nvidia Tesla V100s each with 32GB of video memory. Depending on the hardware and the model architecture, models took anywhere from minutes to train for small models, up to 10+ hours to train for the largest ones.

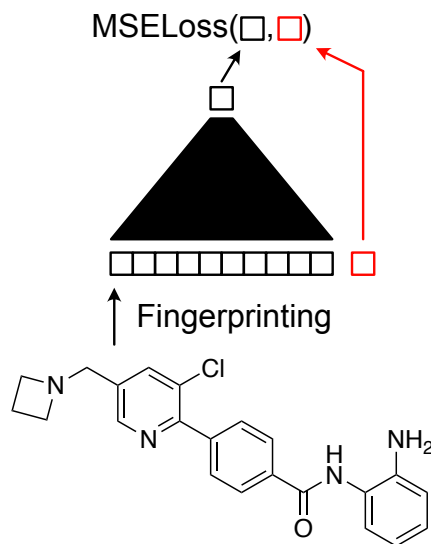


Fig. 2.1. General flow of information for QSAR property predicting using an MLP. Item in red is the true activity, unknown at validation time but known at training time to train the model.

2.3. Baseline Predictor

The main prediction method used in this work is the multilayered perceptron, described in detail in section 1.3.1. The general flow of information for all predictions throughout this entire work is the following; a compounds passes through a fingerprinting mechanism to make a fixed length vector that is passed through an MLP to give a single prediction. This prediction is compared against the ground truth and the network learns from the resulting loss. Figure 2.1 shows this flow of information graphically, which is expanded on in future chapters. This is referred to as the "baseline predictor", "single predictor" or "single approach" because it is used for comparison when the pairwise approach is introduced in chapter 4. However it serves as the only predictor used in the following chapter 3.

2.4. Metrics

All tasks reported in this work are regression tasks and the Pearson correlation R^2 , averaged across 5 cross validation folds, is reported throughout the experiments as the main prediction performance metric. The models are trained with Mean Squared Error (MSE) as the loss function. Mean Average Error (MAE) is also used occasionally to report results alongside Pearson correlation R^2 . Classification tasks and their corresponding metrics are starkly absent from this work because, as will be discussed in section 4, the pairwise approach

is incompatible with typical property classification datasets. These datasets do not have numeric property data for each compound but instead report "hit or not-hit" for bioactivities. Datasets like these represent a large portion of all chemical activity datasets, but are not used in this work.

2.5. Hyperparameters

HyperOpt [7] was used to perform the Hyper-Parameter Optimizations (HPO) for each experiment. Pearson R^2 was used as the target of the meta-optimization. For each fingerprint type and dataset, the optimum value for the remaining hyperparameters were found and used for a final five-fold cross validation (occasionally reduced to three or two fold for large and slow-to-train models). Fingerprint size, number of fingerprint layers (fingerprint radius r), number of MLP layers, size of MLP hidden layers, number of gradient steps and dropout were optimized over the course of 10 runs for each model. Hyperopt was generally needed only for the two graph convolutional network fingerprints, as they are significantly more computationally costly and require a sequential hyperparameter optimization process as a parallel approach is not viable due to memory constraints.

Chapter 3

Reduced Collision Fingerprints (RCFP)

3.1. Motivation

The first contribution of this work is a new kind of fingerprint, based on ECFP, but that seeks to investigate and reduce the influence of bit collisions. Fingerprint bit collisions occur when the hashing process of fingerprint generation produces the same fingerprint bit for different atom IDs. This can conceivably lead to confusion for a predictor. In the generation process, discussed in section 1.2.5 and shown in figure 1.6, atom identifiers are hashed to a 32 bit integer space (2^{32}). It is generally assumed that there would be few collisions in this rather large space for most values of the fingerprint radius r [69, 24]. Some applications can make use of the fingerprints directly in this large space, typically represented as lists or dictionaries of IDs. These applications include measuring similarity and chemical retrieval tasks [80]. For QSAR modelling using predictors like MLP's, a fix-length vector of reasonable length is needed.

For these tasks, the 32 bit identifier of each atom is then "folded" with modulo arithmetic to activate a single bit in the fingerprint space (commonly 2048 bits, or 2^{11}). This massive compression of the possible bits results in the majority of these potentially confusing bit collisions. To make a more information-rich and useful fingerprint, we propose a modification to ECFP referred to as Reduced-Collision ECFP (RCFP) which alleviates these collisions. Therefore, RCFP remains exposed to collisions from the hashing process itself but given the large 32 bit space, these are considered to be unlikely for modest fingerprint radii and databases sizes.

Figure 3.1 shows the effect of bit collisions on fingerprint diversity for standard ECFP on the AurA dataset. We can see that as the fingerprint size is increased along the x axis, the number of unique bits across the entire dataset increases. Every increase in the number of unique bits (y axis) as the fingerprint increases in size (x axis) is a fingerprint collision being resolved. The plateau that these curves reach is when every possible substructure (up to the specified radius) for each of the molecules in the dataset is represented by its own

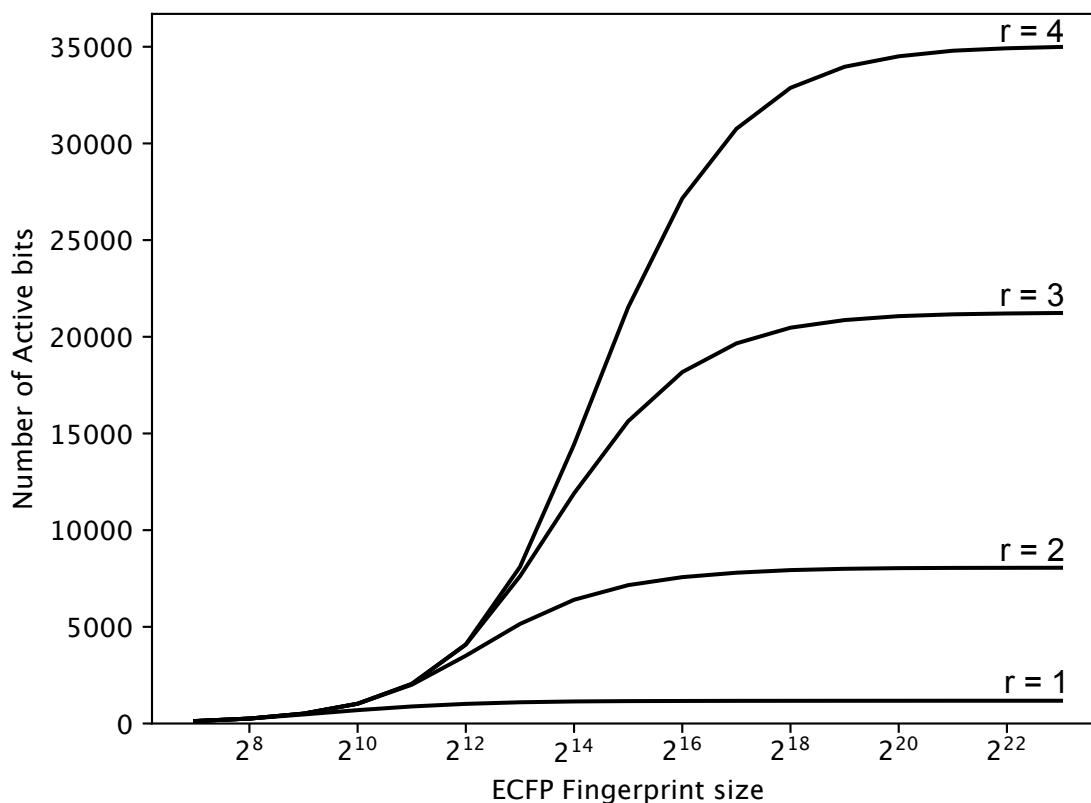


Fig. 3.1. Number of active ECFP bits across the AurA dataset for different values of fingerprint radius r . Increases active bits as the fingerprint grows are fingerprint collisions being resolved.

fingerprint bit. Achieving this level of collision reduction takes very large fingerprint sizes for traditional ECFP, but the number of active bits across an entire given dataset is quite reasonable in comparison to the ECFP size. For example, for radius 4, it takes a fingerprint of around 2^{20} bits to represent around 35,000 unique substructures. This means that 96% of fingerprint bits are never turned on for any compounds in the entire dataset.

RCFP builds a fingerprint based on the unique active bits only. This is done by assigning a fingerprint bit to each unique substructure present across the entire dataset. Therefore, the fingerprint size will be the number of active bits at the "plateaus" shown in figure 3.1. So for the radius 4 example from above, the fingerprint length would be around 35,000. Still quite large, but much smaller than the 2^{20} bit ECFP and representing the same information.

On top of this, not all active fingerprint bits are equally useful. For example, from a learning perspective, a bit has to appear at least twice to be useful in a dataset. It must appear at least once in the training set for the predictor to be aware of it and it must appear

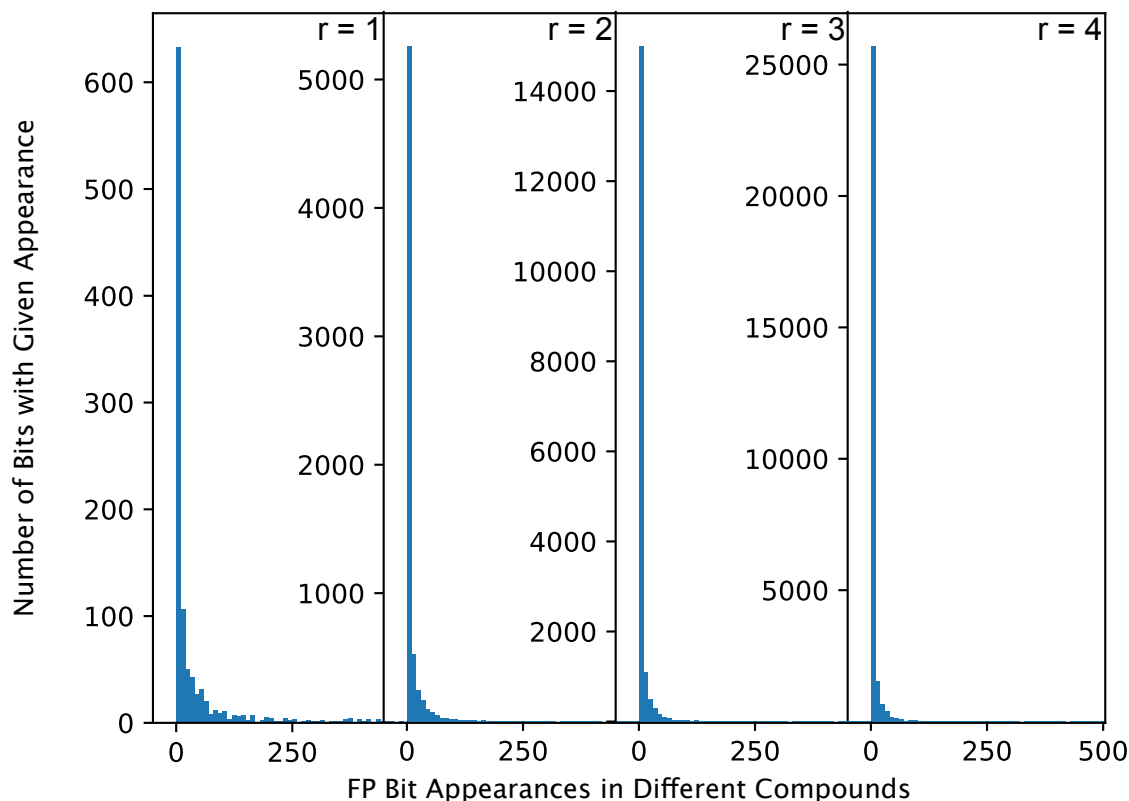


Fig. 3.2. Distribution of Fingerprint uniqueness across the AurA dataset.

again in the validation set to be used for a prediction. It follows that bits appearing more rarely across the dataset are not as useful as bits that are more common. Figure 3.2 shows the distribution of fingerprint bits and their uniqueness in the AurA dataset with various fingerprint radii. We can see that a large population of the unique fingerprint bits appear very few times across the whole dataset.

RCFP also uses this assumption to reduce the length of the fingerprints. As mentioned above, the fingerprint length of RCFP is initially equal to the number of unique bits across the whole dataset. These fingerprints are then "trimmed" based on how many times each bit appears across the dataset. The generation process is explained in detail in the next section.

3.2. Methodology

The RCFP generation process is split into 3 consecutive phases; compiling a fingerprint for the whole dataset, trimming this dataset fingerprint, and using it to make individual compound fingerprints.

For phase one, the database fingerprint is essentially a list of all unique IDs encountered, up to a given radius, across all the compounds in a dataset. This is very similar to the ECFP generation process, except all IDs from different compounds are stored together, along with how many times each ID has appeared. This is shown graphically in figure 3.3 and algorithmically in algorithm 3.2.1. The IDs generated are in the full 32 bit space, and although a hashing algorithm is still applied to concatenated neighbour atom representations, this space is considered so large that collisions will be rare, because the modulo folding step is removed [24].

Algorithm 3.2.1. Dataset FP Generation

```

Input: Dataset D, radius R
  For molecule m in D
    For atom a in m
       $x_a \leftarrow atom\_features(a)$ 
    end For
    For layer l in R
      For atom a in m
         $x_1 \dots x_n \leftarrow neighbours(a)$ 
         $\mathbf{v} \leftarrow [x_a, x_1 \dots x_n]$     % Concatenation
         $x_a \leftarrow hash(\mathbf{v})$ 
         $ds\_ids[x_a] = ds\_ids[x_a] + 1$ 
      end For
    end For
  end For
Return ds_ids

```

The second phase, trimming this database fingerprint, is the process whereby an ID in the dataset ID list is discarded if it has not appeared a required number of times m across the dataset, called the m -cutoff value. This is shown graphically in figure 3.4 and algorithmically in algorithm 3.2.2. This is not to say that the most common bits are the most useful, but that bits appearing more often may be easier to learn from. This is because it will be easier for the predictor to form a general representation based on these bits. This reasoning also applies to IDs that appear in almost every single compound, as there is also not much to be learned from those. The count of each ID is also replaced with its corresponding bit in the final fingerprint. The order of these ids is not important.

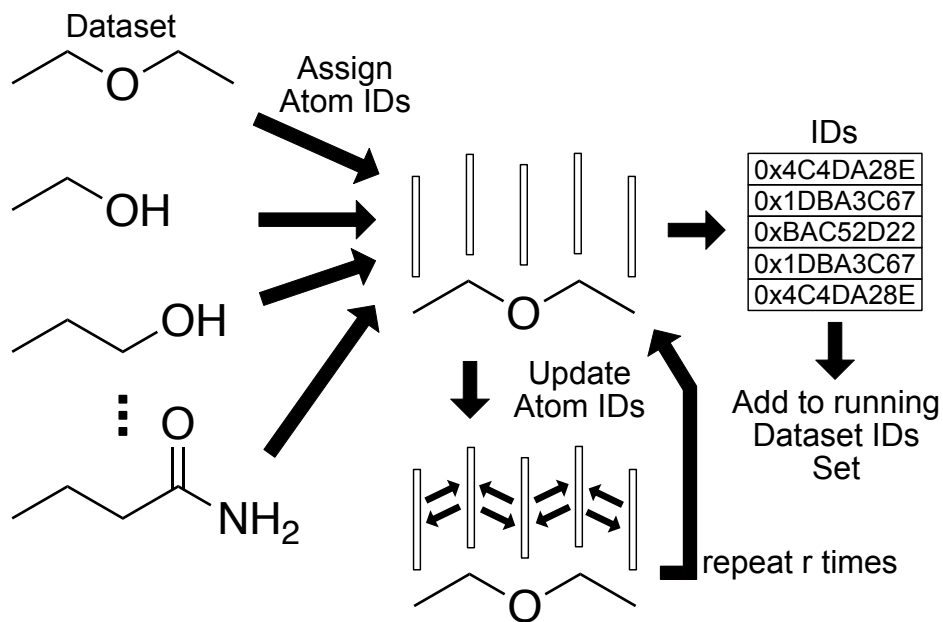


Fig. 3.3. Whole-Dataset or "Master" Fingerprint Generation.

Algorithm 3.2.2. Trim Dataset FP

Input: *Dataset IDs Dictionary* ds_ids , *cutoff* m
counter $\leftarrow 0$
For id *in keys of* ds_ids
 if $ds_ids[id] \leq m$
 remove $ds_ids[id]$
 else
 $ds_ids[id] = counter$
 $counter = counter + 1$
 end if
end For
Return ds_ids

The final phase generates individual RCFP's for each molecule. The trimmed database ID list is used as a key to assign bits in the final fingerprint for each compound. This is shown graphically in figure 3.5 and algorithmically in algorithm 3.2.3.

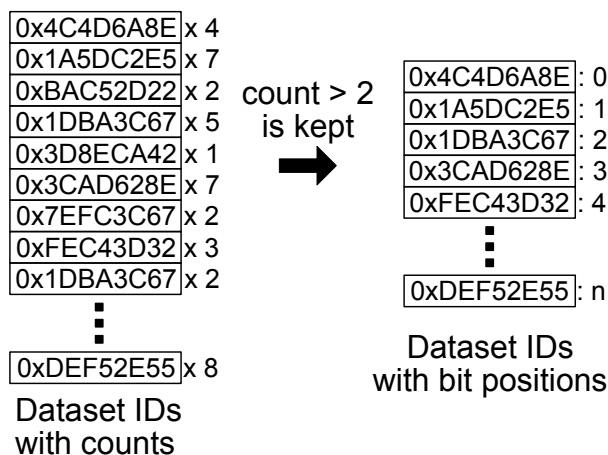


Fig. 3.4. Database Fingerprint Trimming Process.

Algorithm 3.2.3. RCFP Generation

Input: molecule m , radius R , Dataset IDs Dictionary ds_ids

For atom a in m

$x_a \leftarrow atom_features(a)$

end For

$f \leftarrow \mathbf{0}_{|ds_ids|}$

For layer l in R

For atom a in m

$x_1 \dots x_n \leftarrow neighbours(a)$

$\mathbf{v} \leftarrow [x_a, x_1 \dots x_n]$ % Concatenation

$x_a \leftarrow hash(\mathbf{v})$

if x_a in keys of ds_ids

$f_{ds_ids[x_a]} \leftarrow 1$

end For

end For

Return f

The overall size of the fingerprint is dictated by 1) the number of substructures counted per compound, which is the number of fingerprint layers 2) the number of compounds in the dataset, 3) the diversity of the compounds in the dataset, and 4) the m -cutoff value, the required number of occurrences of a given substructure for it to be included in the fingerprint.

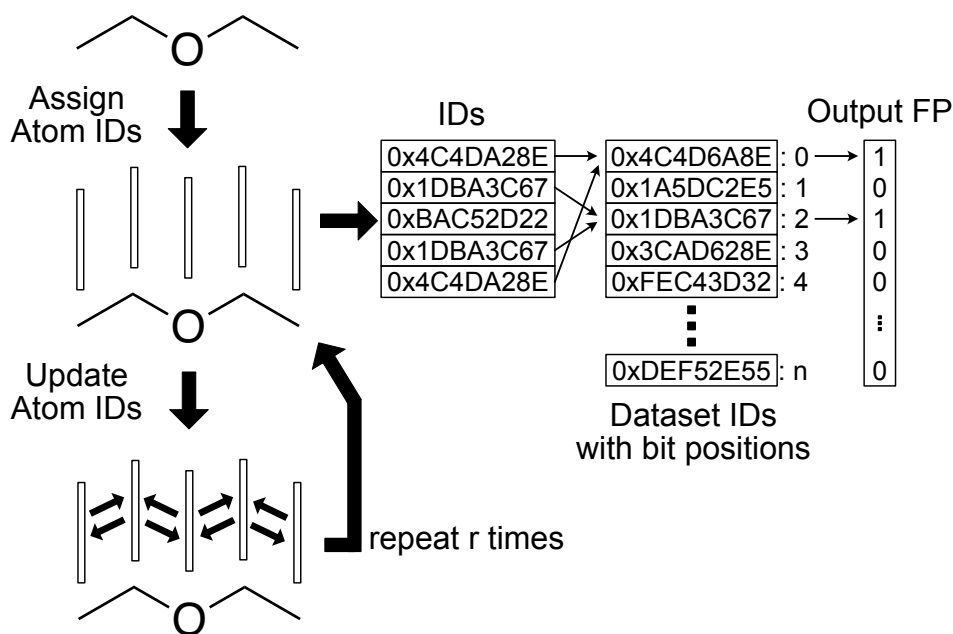


Fig. 3.5. Individual RC-ECFP generation.

m is the only source of control over the length of the fingerprint, so for larger datasets m must be increased to keep the length of the fingerprints reasonable.

3.3. Results and Discussion

3.3.1. Similarity Metrics

Chemical similarity is difficult to quantify because chemicals are amorphous graphs with variable size and connectivity. Because of this, there is no concept of a "ground-truth" for chemical similarity, and every measure is imperfect to some degree. A very common measure of chemical similarity in literature is the Tanimoto similarity of compound fingerprints. Other measures of similarity exist, such as 3D structural alignment of molecules, but Tanimoto distance is more common [54]. The equation for Tanimoto similarity of two bit vectors is shown in equation 3.3.1 and can be generally referred to as the "intersection divided by union", which in the case of binary vectors is the sum of a bit-wise "and" operation divided by the sum of a bit-wise "or" operation. The metric was originally formulated by Jaccard in 1912 [46] and independently by Tanimoto in 1958 [81]. Different fields refer to this metric as either Tanimoto similarity or Jaccard similarity, with cheminformatics tending to refer to it as Tanimoto similarity. In this section, the Tanimoto similarities of ECFP and RCFP for

pairs of compounds randomly sampled from the AurA dataset were examined for fingerprints of the same length.

$$T(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \text{ and } B|}{|A \text{ or } B|} \quad (3.3.1)$$

Figure 3.6 shows the pairwise fingerprint similarity investigation. We can see that RCFP has a higher tendency to identify compounds as being more similar (as measured by Tanimoto similarity) than ECFP will determine for the same pair. This is shown by the higher population above the unity line. This makes sense given that RCFP tends to be slightly denser. This could be due to collisions existing in ECFP but being resolved in RCFP, and if this is the case, then a single bit will become two. If more of the new bits are present in the same two compounds, they now share multiple bits of similarity instead of just one, which is an increase in similarity for the same sized fingerprint. The compound pairs showing 100% similarity in both fingerprints are likely stereoisomers, as neither fingerprinting method takes compound chirality into account. This means that distinct compounds may have the same connectivity and therefore fingerprint.

For the population that exists below the unity line, there is higher similarity in ECFP than in RCFP for a given pair of compounds. This could be because the RCFP fingerprints, which used an m cut-off threshold of 7, happened to remove bits that the compounds shared. In cases where useful bits that contribute to similarity are discarded due to the threshold, ECFP may show higher similarity.

In figures 3.7 and 3.8 we can see that sampled pairs are already unlikely to be similar to each other in the first place, as shown by the density of points in the low-similarity region. Since pairs of compounds tend to be more similar when measured by Tanimoto distance in RCFP, and because the similarity spectrum is bounded, perhaps it can be better utilized with RCFP than with ECFP. Comparing the density of fingerprint differences between the two different fingerprints in the two figures, we can see that going from ECFP to RCFP noticeably shifts the density for the more similar pairs but imperceptibly shifts the density for the less similar pairs. This is a qualitative observation and it is possible that all pairs have simply moved an equal, imperceptible amount so further investigations would be required to confirm or deny this. With more pairs in the "high similarity" area (traditionally defined as similarity of 0.5-0.6 and up) of the difference/activity plots, activity cliffs can be identified more easily. Activity cliffs are pairs of compounds where small changes in structure result in large changes in activity and are important features to discovery [77].

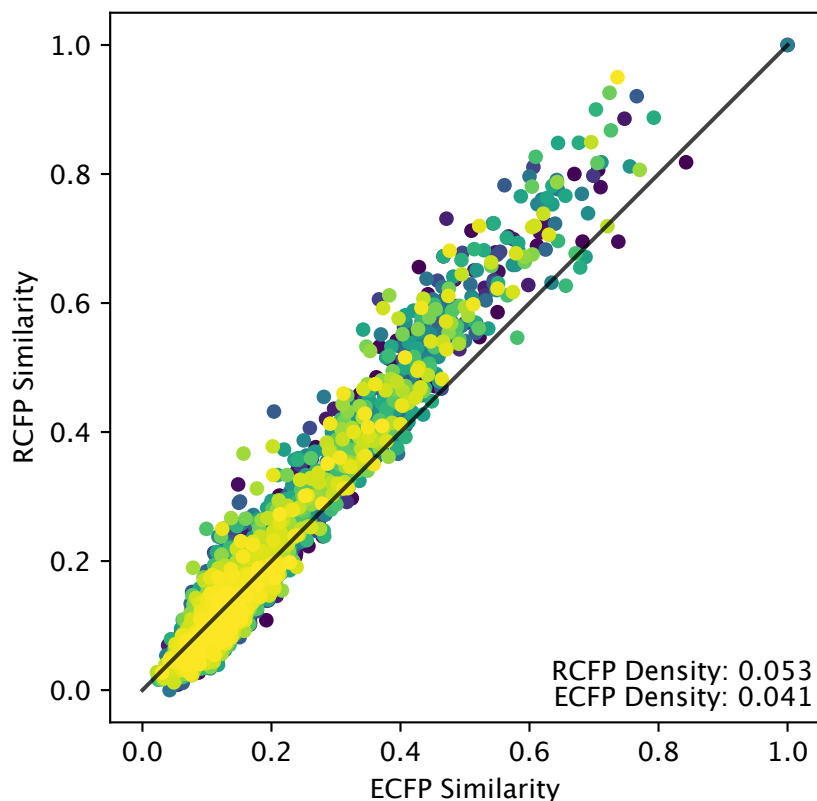


Fig. 3.6. Randomly sampled pairwise similarities from ECFP and RCFP. RCFP of radius 3 with an m -cutoff of 7 was used to make fingerprints of length 3101. ECFP of the same length were made and 10,000 pairs of compounds were randomly sampled.

3.3.2. m -Cutoff

The "necessary number of occurrences" m cutoff value from step two of the RCFP generation process, algorithm 3.2.2 and figure 3.4, is important for controlling the size of the fingerprint. Figure 3.9 shows the size of the fingerprint as a function of the m cutoff value for the three datasets featured in this work. As mentioned above in section 3.2, 4 factors control fingerprint length; fingerprint radius, dataset size, dataset diversity and the m values which trims the least occurring bits. We can see in the figure that fingerprint size is reduced as m increases, especially for the largest of the datasets, CEP. Higher m values result in a larger loss of information, however the hope is that by discarding rarely occurring bits, the information loss is kept to a minimum.

We can see that with the larger CEP dataset, the number of compounds and their diversity make for a very large number of unique substructures and thus very large fingerprints. This makes it clear that there is an upper bound to the utility of RCFP that is based on

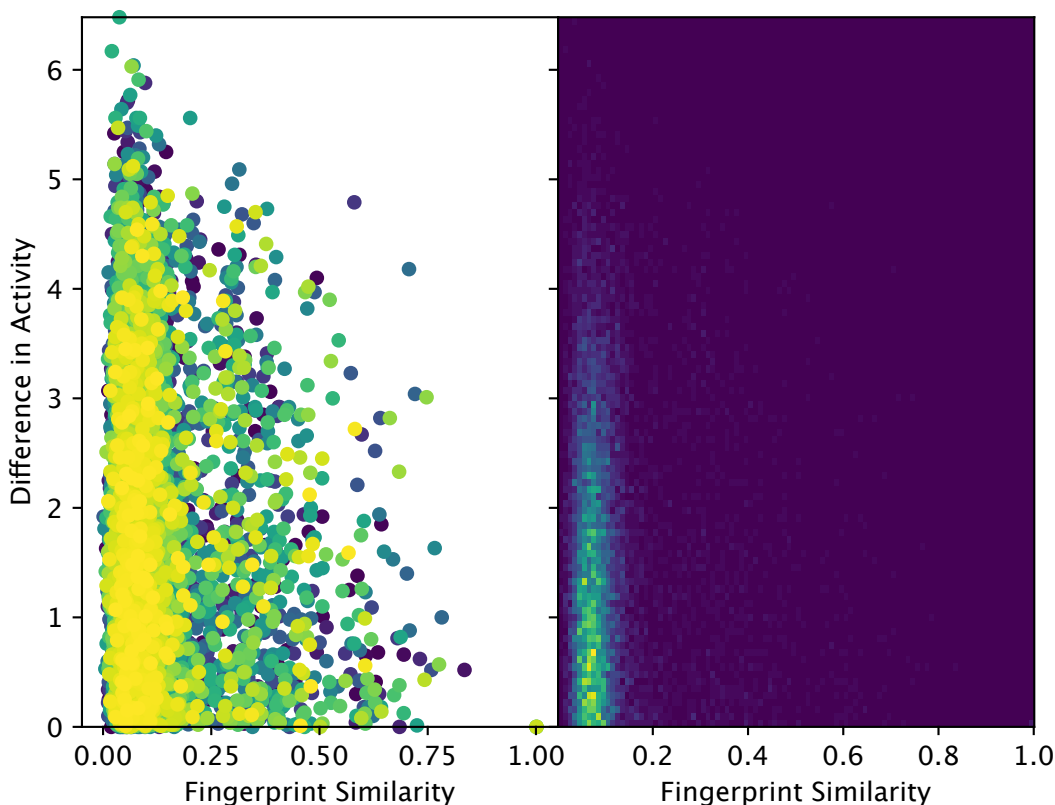


Fig. 3.7. Randomly sampled pairwise ECFP similarities and their activity differences, 10,000 random pairs of compounds. A scatter plot is shown on the left and a 2d histogram on the right.

dataset size/diversity, after which the RCFP approach is simple too cumbersome. Perhaps a dimensionality reduction process like Principal Component Analysis (PCA) would be useful to reduce fingerprint sizes when a very large/diverse dataset is used.

3.3.3. Prediction Performance

The performance of these reduced collision fingerprints as the input to learning models is compared with ECFP in figure 3.10, shown with the circular plot points. The figure shows the prediction performance on the AurA dataset, in R^2 value on the top and in MAE on the bottom. To determine fingerprint sizes, the m -cutoff value for RCFP is varied from 0 to 150 and the resulting fingerprint size of RCFP is also used to make the ECFP values for that point. For example, using an m -cutoff value of 5 results in an RCFP size of 3719 bits, so that RCFP size is compared to ECFP of equivalent size. For these experiments, the number of fingerprint layers is fixed at 3, batch size 1024 and independent hyperparameter

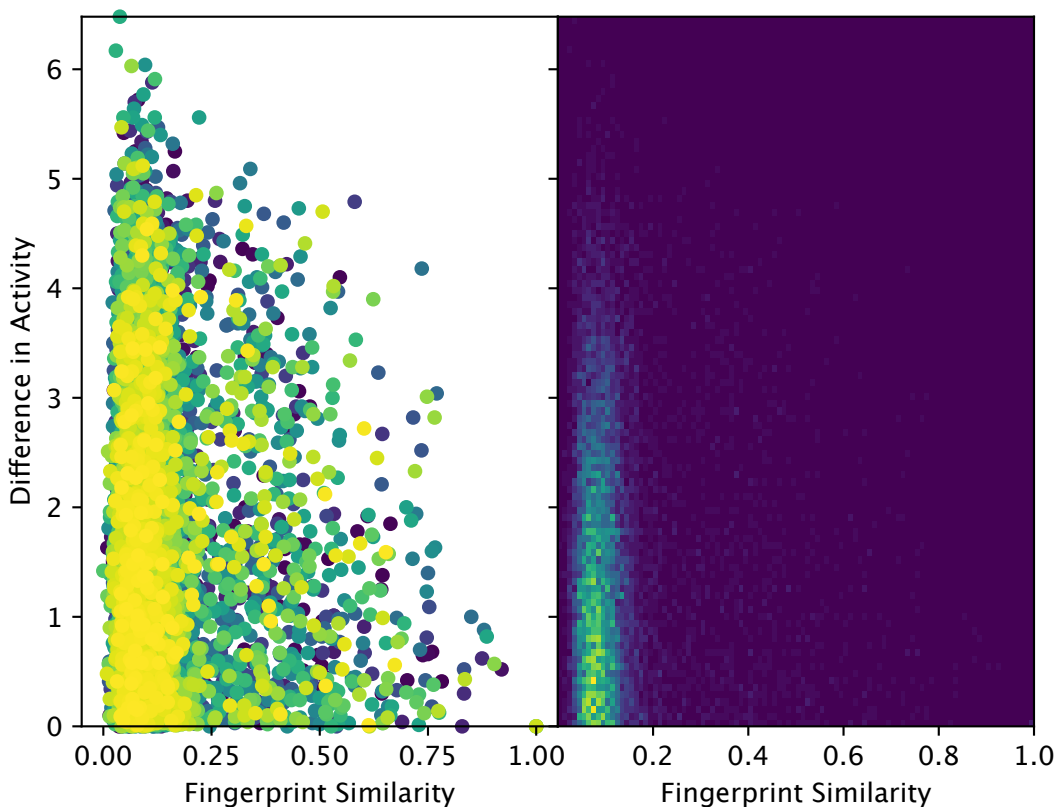


Fig. 3.8. Randomly sampled pairwise RCFP similarities and their activity differences, 10,000 random pairs of compounds. A scatter plot is shown on the left and a 2d histogram on the right.

optimizations were done for each fingerprint type and size to determine the best values for number of MLP layers, MLP layer sizes and MLP network dropout.

We can see that despite arguments for the potentially more informative nature of RCFP, this does not translate to a higher predictive power in an MLP predictor, save for some seemingly arbitrary fingerprint sizes. The reason for this is possibly the ability of the MLP to resolve complicated relationships between input bits means that fingerprint bit collisions are simply not confusing to the MLP predictor. The high capacity, many parameter nature of the method can sort through the confusing bit contributions. On top of this, RCFP removes information from the fingerprint when an m -cutoff value of greater than 0 is used. This loss of information is likely detrimental to the prediction performance and is not made up for by the collision-free nature of RCFP.

If we reduce the capacity of the models, we can remove the predictor’s ability to sort through the confusing bit collisions. To explore this, we use a linear model, as it is simply

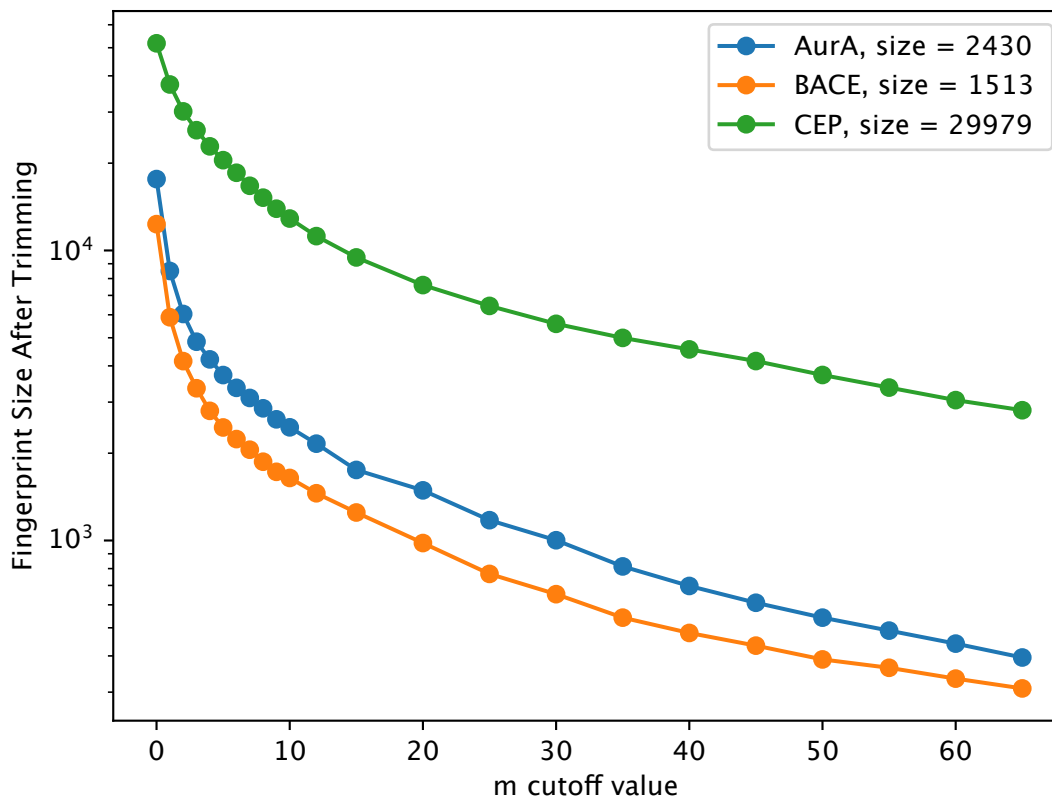


Fig. 3.9. Fingerprint size after trimming with progressively larger m cutoff value for different datasets.

an MLP with no hidden layers and significantly lower predictive capacity. The results are shown on the same figure, 3.10, with the square plot points. The fingerprint size varies as before, but with 0 hidden MLP layers and 0 network dropout to make it a linear model. Early stopping is used to prevent overfitting and the optimal number of training steps varied widely between fingerprint sizes, from 140 steps up to over 1,000 depending the fingerprint type and size.

We can see that when using a predictive model with a weaker representation capacity, RCFP performs better than ECFP at most fingerprint sizes. This could be due to the removal of collisions making up for the loss of information from the m -cutoff trimming process. One potential advantage of using linear regression over a higher-capacity MLP predictor could be explainability. In a linear model, each weight corresponds to the relative importance of that bit to the prediction. For RCFP, these bits are also unique substructures through the established one-to-one fingerprint bit to substructure relationship described in section 3.2. This one-to-one relationship is one directional due to being a hash function. However, in a

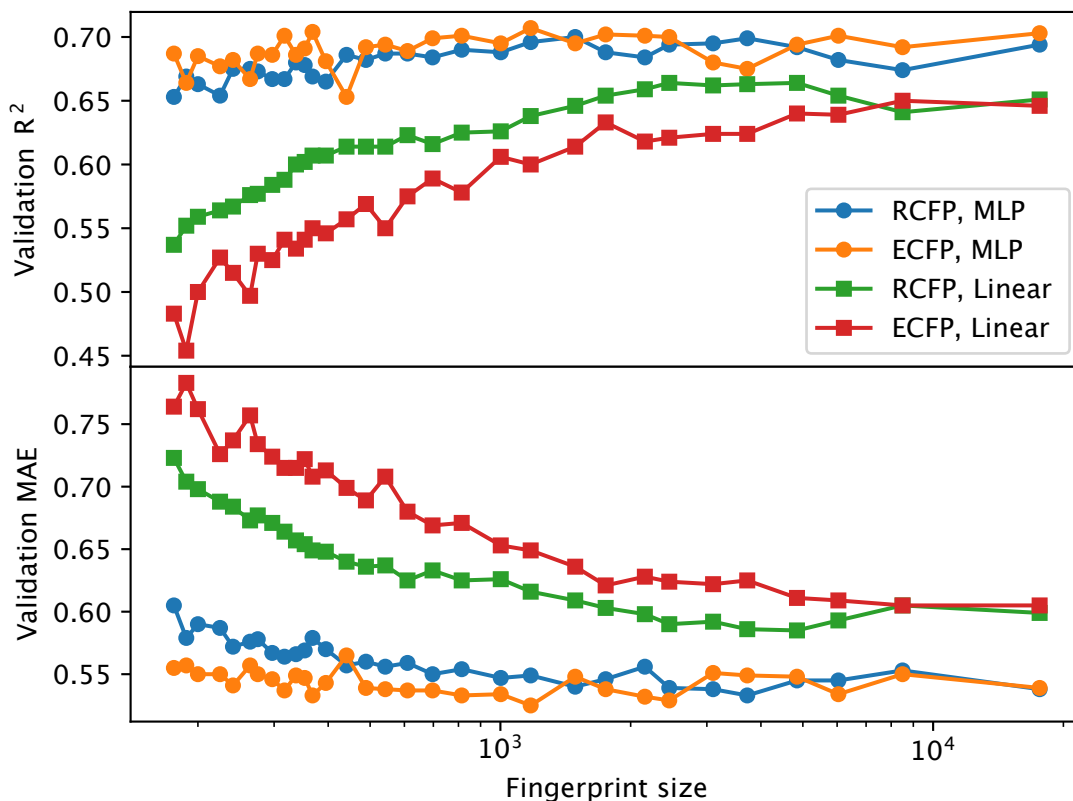


Fig. 3.10. Performance of both RCFP and ECFP in MLP and Linear predictors, for different fingerprint sizes (determined by varying m -cutoff values) on the AurA validation dataset. Higher is better for R^2 and lower is better for MAE.

situation with the entire training set available, the hash function can be reverse engineered via process of elimination to map the reverse relationship from bits to substructures. Therefore, using RCFP in this way can make a better direct link between compound substructures and their importance to the predictions. This is the idea of intrinsic explainability in RCFP, that the identity of each substructure could be deduced from maintaining the trimmed database ID list obtained from phase 2 and using it in reverse.

A possible concern for RCFP generation is the idea of increasing the size of the dataset after the fingerprints have been generated. If a given dataset were to be expanded in the future, RCFP can deal with this in several ways. First off, in the case of inference, there is no need to account for new unique substructures of the inference compounds because their new unique bits will not be present in the training set, so the model will not be trained on this information anyway.

For new training compounds it is evidently more complicated. All RCFP's are changed if even a single compound with a single new unique substructure is added to the training

set. However, every unique structure is accounted for with a bit in the previous (pre-new compound) fingerprint. This means whatever new bits are assigned for the new training compounds, they will not be present in the old compounds. Although the fingerprints of the old training compounds will technically become larger, they will only be padded with zeroes. The new fingerprints will have bits occupying both the old space and a number of new bits that represent structures unaccounted for by the old fingerprints. These new bits could be accounted for separately. The models trained on these fingerprints will need to be retrained as the fingerprint size has changed. A possible solution to this could be using the already-trained "old model" in tandem with a new model trained on the "new fingerprint" and aggregating the results of the two models, but this was not investigated in this work.

Our motivation for developing RCFP was the possible confusion brought on by the collision-prone ECFP fingerprinting method. RCFP was built around avoiding collisions as much as possible and to facilitate this goal, some information had to be removed from the representation. The prediction performance investigations revealed that for an MLP predictor, this trade-off is not a worthwhile one, whereas for it does appear to be for a linear predictor.

Chapter 4

Pairwise Compound Comparisons

4.1. Predictor Descriptions

The final contribution of this work is the pairwise compound comparison approach to property prediction. The motivation for this approach lies in MMPA, which is common in drug discovery and is introduced in section 1.4.3. The fundamental unit of data for the pairwise approach is a pair of molecules, which contrasts with standard QSAR models where each data point is a single molecule, referred to as the single approach. The prediction performance of the two approaches is compared in this chapter. The fundamental predictions made by models of the pairwise approach concern the relative property values of the pair. For the regression task, the predicted value is the property difference between the two compounds (can be positive or negative). For the classification task, the prediction is the probability that one compound has a higher property value than the other. The regression task is the main focus of this work and the classification task is mentioned but not examined.

Because the predictors are multi-layer perceptrons, each pair of molecules needs to be represented as a fixed-length vector as discussed in section 1.3.1. Each compound in a pair is represented by one of several molecular fingerprints, and their resulting fingerprints are concatenated to form a single vector that is passed to the MLP that makes the prediction. Figure 4.1 graphically shows the flow of information for the pairwise compound comparison predictions.

The predictions that are made are directional, either the difference in property (positive or negative) when moving from compound A to compound B for the regression task, or the probability that compound B will have a higher property value than compound A, for the classification task. The prediction is compared to the ground truth in a loss function and the gradient of the loss value with respect to the model weights is used to update the predictor. The general framework for this optimization is described in detail in section 1.3.1.

For a dataset of N compounds, there are N^2 possible directional compound pairs that exist. If self-similar pairs are removed this drops to $N(N - 1)$. This number is cut in

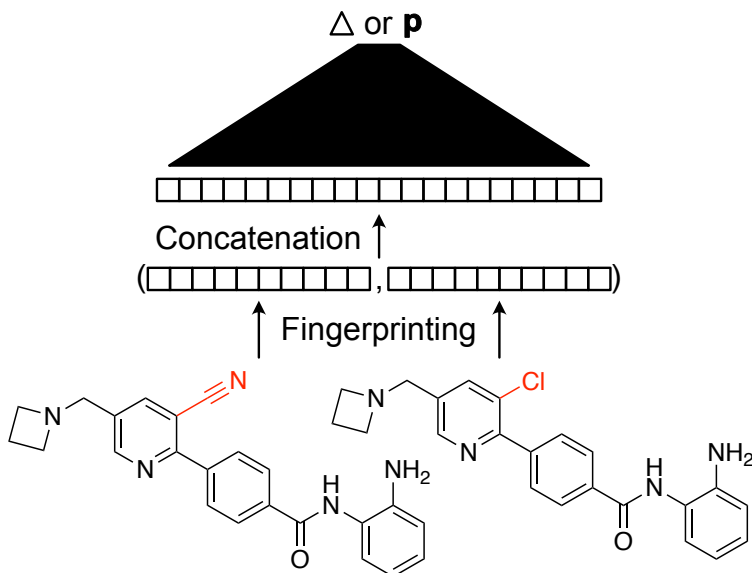


Fig. 4.1. Basic flow of information from a pair of compounds to a pairwise prediction. The final prediction is either the difference in activity (Δ) or the probability of a higher property value (p).

half for non-directional pairs. The underlying dataset is split into two sets, a training set and a validation set, as is typical for machine learning tasks and discussed in section 1.3.1. This divides the pair space into 4 different kinds of pairs as shown in figure 4.2. The first type is pairs of two training compounds, the second and third types are composed of one training compound and one validation compound, and the final type is pairs of two validation compounds. As discussed in section 2.1, some machine learning methods split the dataset into three parts; a training set, a validation set and a test set. In this situation, there would be a more complicated pair landscape but it would still be a natural extension of what is described in the two part split.

The different pair types in figure 4.2 allow for two different validation scenarios. First, models must always be trained on training-training pairs, since both activities must be known to get a proper ground truth for the learning process. For validation, however, the model can use either training-validation pairs (referred to as the TV validation type) or validation-validation pairs (referred to as the VV validation type). The difference between these two validation types results in a different kind of prediction. For the VV validation type, the prediction is the difference in activity, with the model knowing neither of the true activities at validation time. These predictions have a mean of 0 and can be positive or negative, based on which compound is higher in activity. For the TV validation type, the model always has

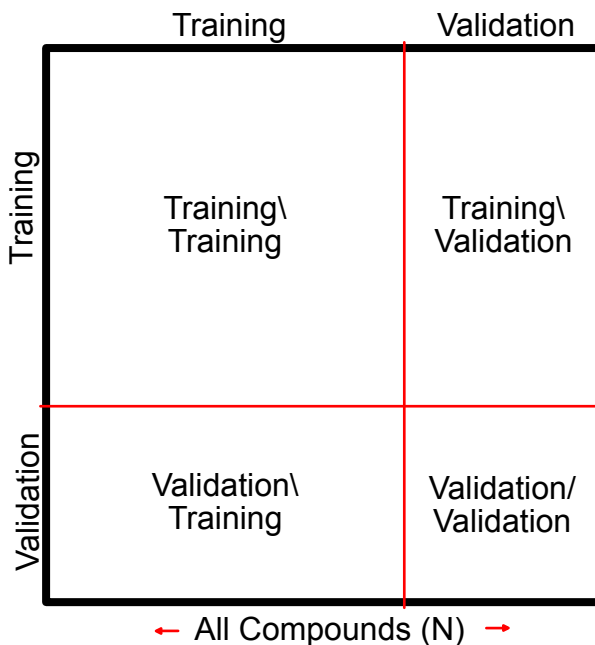


Fig. 4.2. Possible pair types for a dataset split into training and validation compounds.

access to the activity of one compound (the training compound) at both training and test time, so it makes sense to incorporate this activity into the final prediction. The fundamental prediction is the same, a directional difference in activity, but this is directly added to the known activity of the training compound, which makes the value a prediction regarding the activity of the validation compound. The combination of prediction task (regression or classification) and validation type (training/validation or validation/validation) produces the four different scenarios, three of which are shown in figure 4.3.

The fourth option is not shown, the classification task with TV validation type. This approach is not as easy to formulate as the others and was thus not explored. One possible approach would be to make a classification prediction, then incorporate the known property value into the probability, producing a hybrid prediction along the lines of "the probability p that the property value will be higher than x ", where x is the training compound activity. This probability-activity hybrid value could be compared against the ground truth activity for the validation compound and a specially crafted loss function would be needed to calculate the loss.

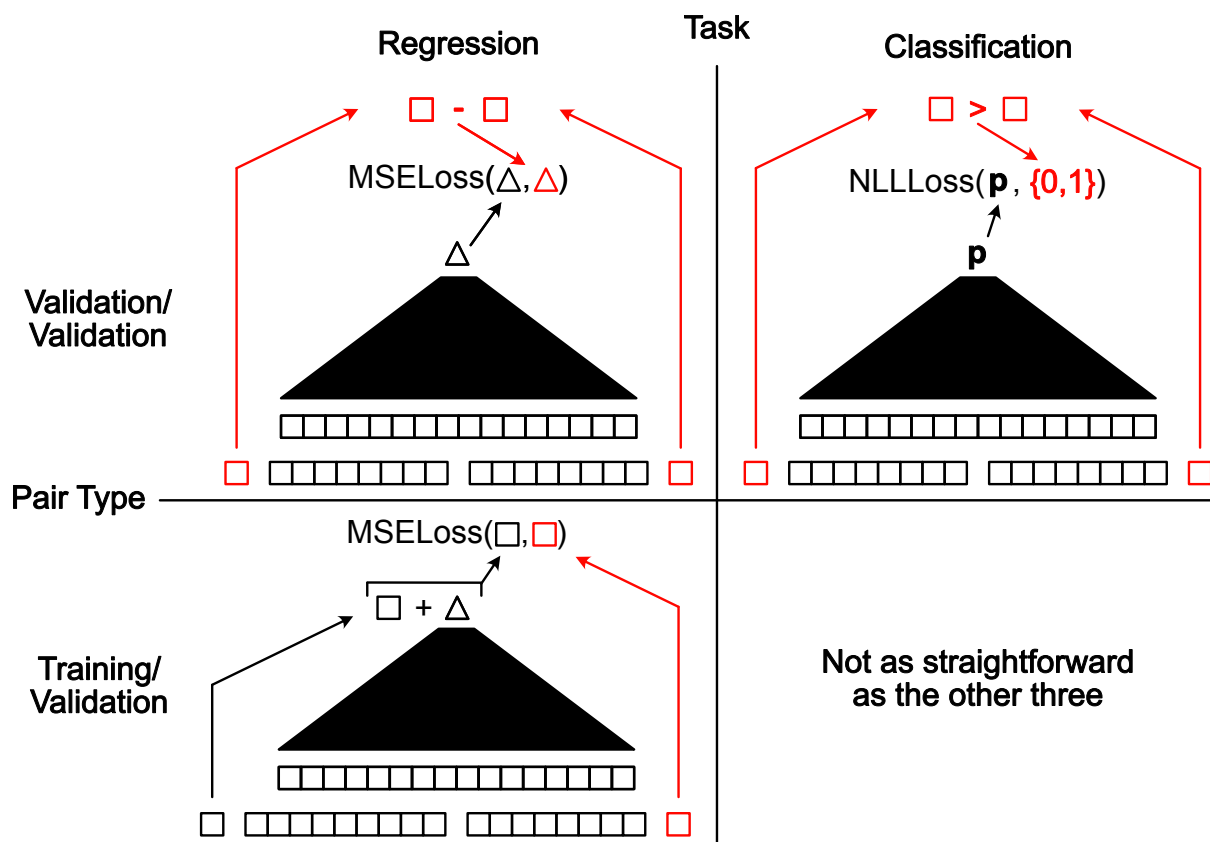


Fig. 4.3. Predictor structures for different tasks and validation types. Items in red are known at training time and used to train the model, but are unknown at validation/inference time. Compounds are represented as vectors by fingerprinting.

4.2. Fingerprint Types

The fingerprinting step in the pairwise predictor (figure 4.1) has the simple requirement of needing a fixed length vector for each molecule. This means it is invariant to the method used to vectorize the molecules. Because of this, several fingerprints types are used; ECFP [69], RCFP, Neural Fingerprints [26] and Chemprop Fingerprints [97].

4.2.1. Fingerprint Type Descriptions

4.2.1.1. Extended Connectivity Fingerprints (ECFP). The first fingerprint approach used is the baseline, the Extended Connectivity Fingerprint described in section 1.2.5. This fingerprint type has seen very widespread use across cheminformatics and is excellent for representing diverse graph structures in a finite fingerprint vector.

4.2.1.2. Reduced Collision Fingerprints (RCFP). The fingerprints described in chapter 3 are used. The m -cutoff value was varied based on the dataset, using 20 for AurA and BACE and 60 for CEP. Figure 3.9 shows the corresponding fingerprint lengths for these values of m . The aim was to select a value for m that results in fingerprints with lengths in the allowed ranges for the other approaches, which use the hyperparameter optimizer to select a fingerprint size between 512 and 4096 bits.

4.2.1.3. Neural Fingerprints. The first of the two learnable/GCN fingerprints used are Neural Fingerprints, re-implemented according to Duvenaud et al. 2015 [26]. Neural fingerprints are meant to be an analogous replication of ECFP, but substituting the non-differentiable functions for differentiable ones and learning the entire process. The process follows the general Message Passing Neural Network framework described in section 1.3.5 and consists of a message phase, an update phase and a readout phase.

The message function, shown in equation 4.2.1, is a simple concatenation of the incoming atom and bond features. The update function, shown in 4.2.2, multiplies the incoming messages from the message function by a certain weight matrix according to the current timestep t and the degree (number of neighbours) of the atom to be updated. The readout function, shown in 4.2.3, takes each node representation, at each timestep, and multiplies it by an output weight matrix that transforms the node representation size to the final fingerprint size.

There are different output matrices for each time step. The result from multiplication with these matrices is passed through a softmax function to ensure that each node at each timestep contributes a total mass of 1 to the final fingerprint, similar to the hash function of ECFP which turns on a single bit for each node representation. The softmaxed values of all the node representations across all the different timesteps are added to form the final fingerprint representation. The form of the message, update and readout functions is shown in equations 4.2.1, 4.2.2 and 4.2.3

$$M(h_v^t, h_w^t, e_{vw}) = (h_w, e_{vw}) \tag{4.2.1}$$

$$U_t(h_v^t, m_v^{t+1}) = \sigma(H_t^{\text{deg}(v)} m_v^{t+1}) \tag{4.2.2}$$

$$R(\{h_v^T | v \in G\}) = \sum_{v,t} \text{softmax}(W_t h_v^t) \tag{4.2.3}$$

4.2.1.4. Chemprop Fingerprints. The last fingerprint type and second GCN fingerprint used in this work are Chemprop fingerprints, re-implemented according to Yang et al. 2019 [97]. Chemprop fingerprints are a more recent, state of the art prediction method that

were developed to alleviate several problems with message passing frameworks. The main difference is that Chemprop propagates edge information between edge labels in a graph, instead of strictly node information. This solves a specific GCN problem where, for node-based message propagation, if in one step a message is passed from node A to its neighbour node B, then in the next step, information will be passed right back from node B to node A again. In a way, A will be updated with information as if it is connected 2 bonds away from itself, which is unnecessary and confusing. Edge propagation removes this problem.

The message function, shown in equation 4.2.4, shows how the messages destined for each edge representation are aggregated. The update function, shown in 4.2.5, is a simple weight matrix multiplied by the message, then added to the original representation of the edge. The readout function, shown in 4.2.6, is a sum of all of the node representations.

$$M(h_v^t, h_w^t, e_{vw}) = \sum_{k \in \{N(v)-w\}} k_{kv}^t \quad (4.2.4)$$

$$U_t(h_v^t, m_v^{t+1}) = \tau(h_{vw}^0 + W_m m_{vw}^{t+1}) \quad (4.2.5)$$

$$R(\{h_v^T | v \in G\}) = \sum_{v \in G} h_v \quad (4.2.6)$$

Because this method updates and propagates edge representations, and the readout function relies on node information, two steps are applied to get the final node representation, ahead of the readout phase. First, for every node, its adjacent edge representations are summed to make a node message vector, shown in equation 4.2.7. Second, this node message vector is concatenated with the original node representation and multiplied by a weight matrix, giving the final node representations, shown in 4.2.8.

$$m_v = \sum_{w \in N(v)} h_{vw}^T \quad (4.2.7)$$

$$h_v = \tau(W_a \text{cat}(x_v, m_v)) \quad (4.2.8)$$

4.3. Train-Validation (TV) Pairs

For the TV validation type, one compound in the pair always has a known property value, even at validation time. This is because one of the compounds in the pair is a training compound. The other compound is a validation compound with unknown property value, referred to as the "query compound". The output of the predictor is a direct prediction of the property of the query compound through addition, as shown in the bottom left panel of figure 4.3. For a given query compound, multiple unique predictions regarding its property

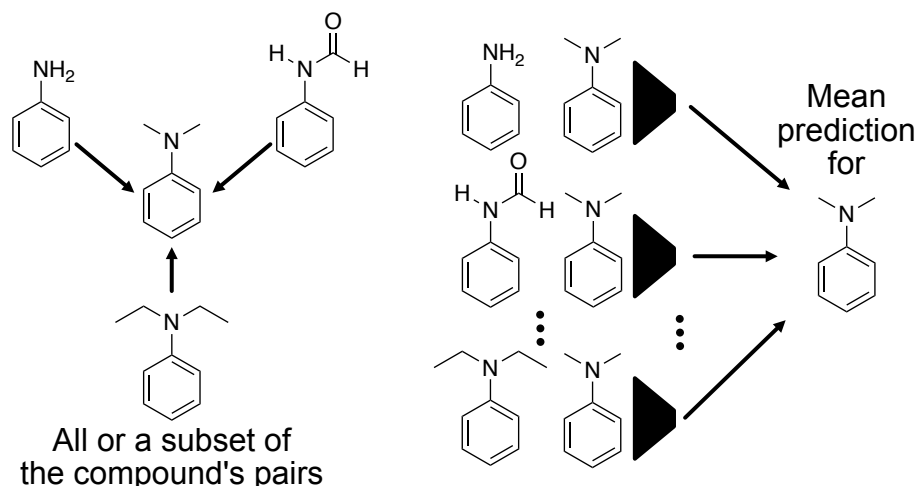


Fig. 4.4. Averaging training compounds to make a final activity prediction from multiple pairs in the TV validation case.

value can be made, as many as there are pairs between the query compound and different training compounds. The final prediction for the property of a query compound can then be a consensus of these unique predictions as is shown in figure 4.4.

When the number of pairs a compound has is very large, the mean of a subset of its pairs can provide a good average prediction for the query compound's property. The number of pairs used is whichever is smaller between all of the possible pairs for a query compound and 150, which provides a good estimation in cases where a query compound has many possible pairs.

4.3.1. Prediction Distributions

Because there are many possible predictions to be made for each compound, the mean is used to access a simple consensus between all the available predictions, as mentioned above. Figure 4.5 shows the full distribution of predictions for both the best (top rows) and worst (bottom rows) predicted compounds, as predicted by a fully trained model on the AurA dataset. The model used ECFP fingerprints of length 3584 with 3 fingerprint layers. The MLP is one hidden layer with 1024 hidden nodes and dropout probability of 0.159, all found by hyperparameter optimization.

We can see for good as well as for poor predictions, the group of predictions is roughly normally distributed. We can infer from this that in both cases, the mean appears to be a good aggregation of the individual predictions. It appears as though no added information can be obtained from examining the distribution of individual predictions, so the mean is

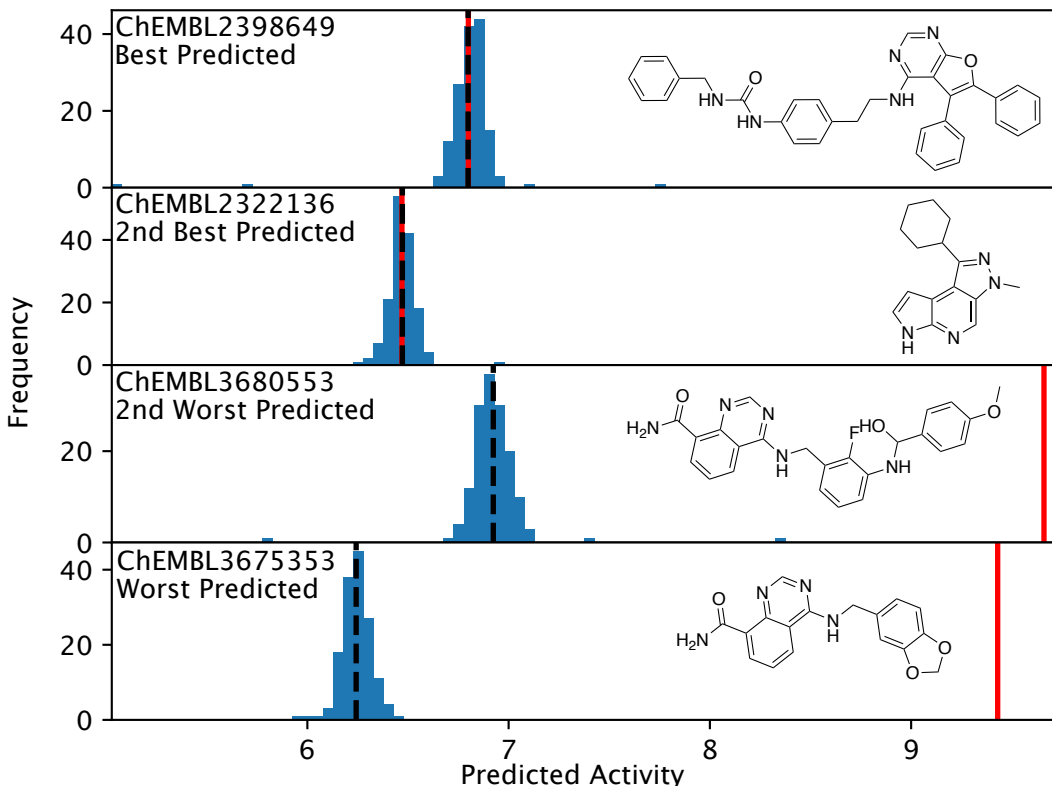


Fig. 4.5. The distribution of activity predictions for the query compounds with the most and least accurate predicted activities from the AurA dataset. Black dotted line is the mean of the individual predictions, the red line is the ground truth activity for the compounds.

reported as the final prediction going forward. These final predictions are what is reported and discussed in the following sections for each of the TV models. Therefore, for TV experiments, aggregate metrics such as R^2 represent an aggregate over all compounds, where the value for each compound is a compound-wise mean prediction.

4.3.2. Repeated Query Compounds

Since the final prediction for a query compound is the mean of many predictions, it makes sense to train the model in a similar fashion. To match the idea of query compound averaging with the training scenario, we consider instead of training on randomly sampled single pairs of compounds (training, query), we train on groups/clusters of pairs of compounds (training₁, training₂, ..., training_n, query). This approach necessitates building a loss function that works on groups of training compounds being paired with a single query compound. First, we consider the loss function for a single pair in equation 4.3.1. Subscript tr is the training compound, subscript q is the query compound. The query compound in this case is of course

just another training compound because we are discussing the training process, but it mimics the eventual query compound in the validation phase, which will be a validation compound.

$$Loss(x_{tr}, y_{tr}, x_q, y_q) = ((\Delta_{pred} + y_{tr}) - y_q)^2 \quad (4.3.1)$$

Expanding this loss function to a group of training compounds all paired with a single query compound, we get the form shown in equation 4.3.2. The mean is used to aggregate the many different pairs because that is what is used at validation time, as seen in figures 4.4 and 4.5.

$$Loss(x_{tr1}, y_{tr1}, \dots, x_{trN}, y_{trN}, x_q, y_q) = \frac{1}{N} \sum_i^N Loss(x_{tr\ i}, y_{tr\ i}, x_q, y_q) \quad (4.3.2)$$

Stepping back to equation 4.3.1, we apply this loss to an entire batch of pairs at a time. This is how the method is implemented and is more efficient and practical. This batch-version of equation 4.3.1 is shown in equation 4.3.3, where BS is the batch size.

$$Loss([x_{tr1}, x_{q1}, y_{tr1}, y_{q1}], \dots, [x_{trBS}, x_{qBS}, y_{trBS}, y_{qBS}]) = \frac{1}{BS} \sum_j^{BS} Loss(x_{trj}, y_{trj}, x_{qj}, y_{qj}) \quad (4.3.3)$$

Applying this same idea to the "group of pairs" loss function, equation 4.3.2, the following is obtained. Batch size in this case becomes the number of training groups in the batch, so the compound level batch size is now $(BS)(N)$

$$Loss([x_{tr1}, y_{tr1}, \dots, x_{trN}, x_{trN}, x_q, y_q]_1, \dots, [x_{tr1}, y_{tr1}, \dots, x_{trN}, x_{trN}, x_q, y_q]_{BS}) = \frac{1}{BS} \sum_j^{BS} \left(\left[\frac{1}{N} \sum_i^N Loss(x_{tr\ i}, y_{tr\ i}, x_q, y_q) \right]_j \right) \quad (4.3.4)$$

Now we can slightly simplify the double mean, if we keep in mind that i refers to the index within a training group and j refers to the index of that training group within the training batch.

$$Loss([x_{tr\ 1}, y_{tr\ 1}, \dots, x_{tr\ N}, x_{tr\ N}, x_q, y_q]_1, \dots, [x_{tr\ 1}, y_{tr\ 1}, \dots, x_{tr\ N}, x_{tr\ N}, x_q, y_q]_{BS}) = \frac{1}{(BS)(N)} \sum_j^{BS} \sum_i^N Loss(x_{tr\ ij}, y_{tr\ ij}, x_{qj}, y_{qj}) \quad (4.3.5)$$

Using this loss function we can train the model on mini-batches composed of groups of training compounds each with a query compound. In practice, we can use the single pair loss function 4.3.1 and its batch-wise counterpart 4.3.3, but simply repeat the query compound several times in different pairs. This is done for each training compound of that query compound’s group. Since the loss is already being averaged over the mini-batch, this also averages over a query compound’s training group members.

The number of training compounds in each query compound’s group was varied and the performance differences are reported in figure 4.6. This number of training compounds is referred to as the "training repeat" number due to how in practice, as mentioned above, the query compound is repeated within the batch for different training compounds. For this experiment, the number of query compounds per batch is fixed, meaning the batch size changes with each number of repeats. Each network was trained to a stable minimum, using early stopping when necessary to compensate for the different batch sizes.

We can see that there is an immediate gain in performance as the number of training repeats increases, but no large gains after increasing to 10-15 repeats. From these results, we set the training repeat number to 16 going forward for all TV experiments. Occasionally, it is lowered to 8 when the model being trained is very large and the batch size needs to be lowered to stop Out of Memory Errors.

4.3.3. Performance Compared to Single Prediction

The performance of the pairwise TV approach is compared to the single approach, with results shown in figure 4.7. Performance is presented in R^2 for the three datasets and all four of the discussed fingerprints. Each bar represents an individual hyperparameter optimization for that fingerprint type, dataset and approach (pairwise vs single) as described in section 2.5. Each HPO step was conducted for 15,000 training gradient steps for the single approach and 25,000 steps for the pairwise TV approach.

We can see that there appears to be no improvement in prediction performance when using the pairwise TV approach compared to the single approach. Regardless of what fingerprint is used, the paired input and prediction averaging approach discussed above does

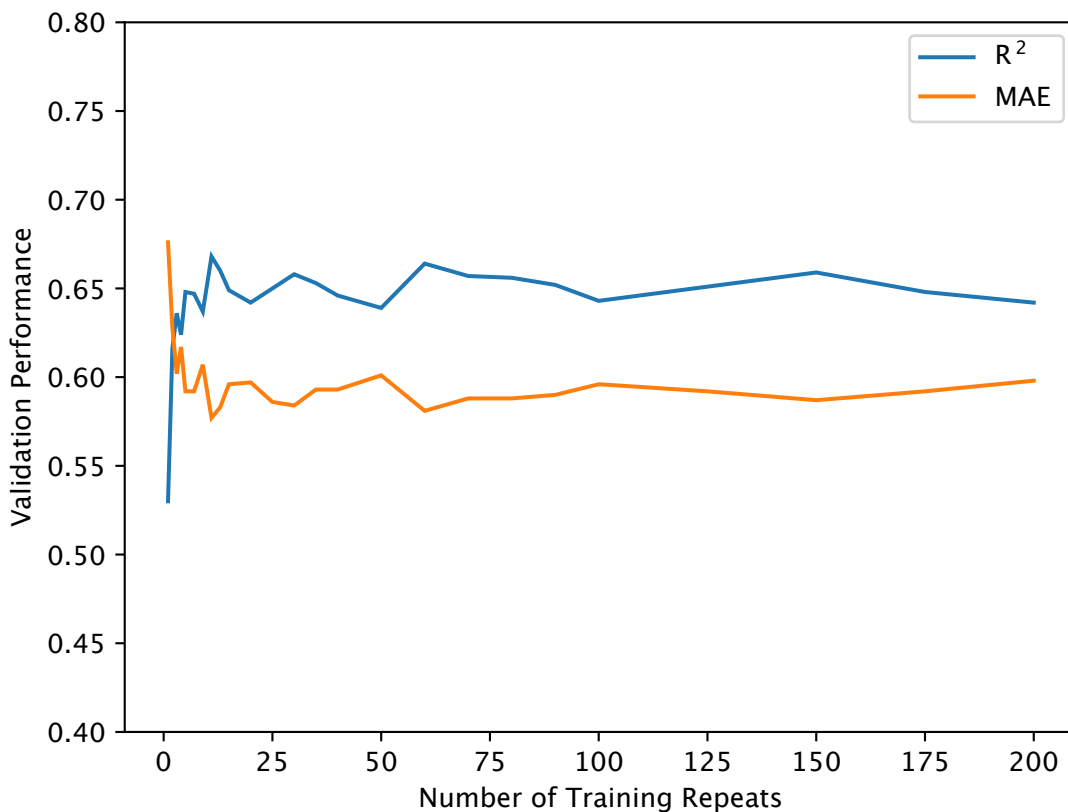


Fig. 4.6. Performance in R^2 and MAE resulting from modifying the number of compounds in the training group of each query compound, called its "training repeat" number. For R^2 higher is better and for MAE lower is better.

not provide consistently better performance on any dataset by a noticeable margin. This raises the question of whether the model is actually learning any pairwise information or just focusing on the query half of the input, which is discussed below in section 4.3.4.

For the CEP dataset with the GCN fingerprints (Neural FPs and Chemprop FPs) we see the largest difference in performance between the pairwise and single approaches, with the single approach performing better. This could be because these models are among the largest in terms of parameters and are trained on the largest dataset. Perhaps they need special consideration or a higher number of iterations than the rest.

Also worth noting is the comparison between the performance of the two GCN fingerprints (Neural FPs and Chemprop FPs) compared to the two algorithmic ones (ECFP and RCFP) across the datasets. The GCN fingerprinting approaches are expected to be more performant, but here we see that is not always the case, particularly for smaller datasets such as AuraA and BACE. We see the largest gains from GCN fingerprints for CEP, which is the largest

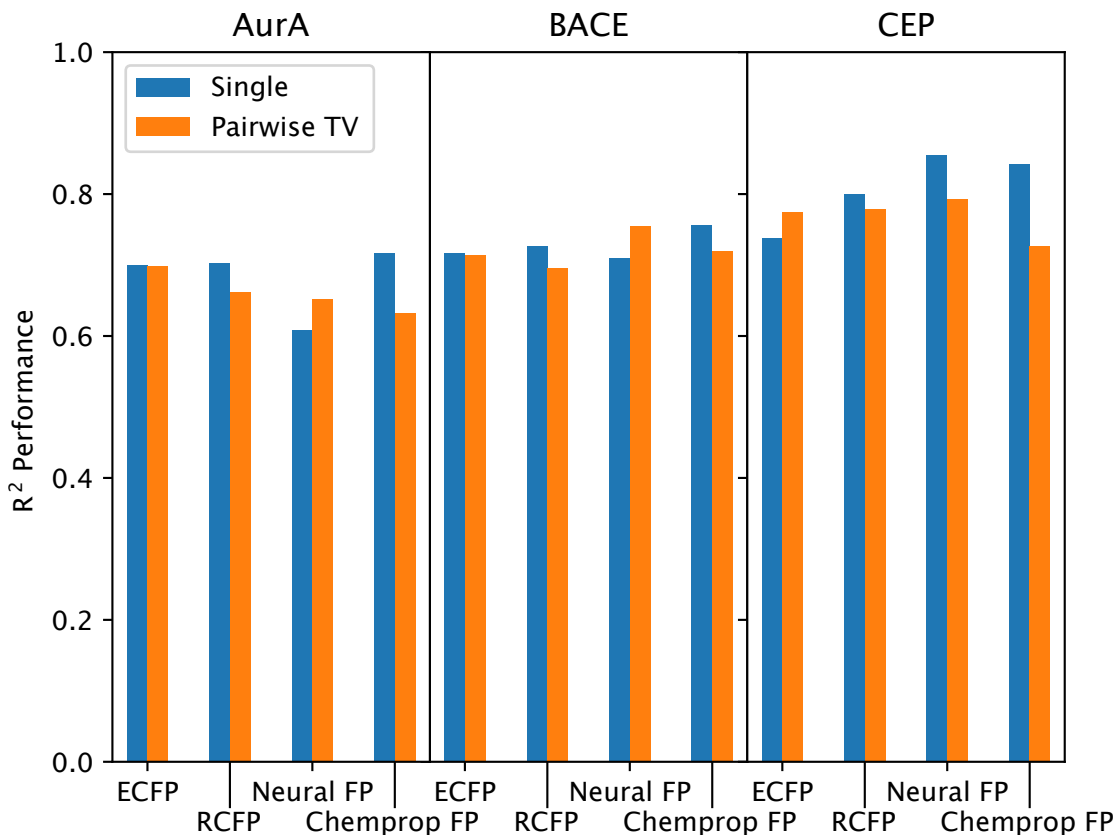


Fig. 4.7. Performance comparison between the pairwise TV approach and the single approach. Measured in R^2 for several fingerprint types and datasets.

dataset. This is understandable given that models with more parameters will train better on larger datasets.

4.3.4. Query-Only Prediction

There seems to be no improvement in performance of the Training/Validation pairwise approach in comparison to the single approach, which begs the question of whether the model gains anything from having pairs as input. One could imagine a situation where the MLP model learns to focus on only the right half of the input, the query compound, to make its prediction and ignores the training compound half of the input.

To test for this behaviour, we devised a query-only experiment, where the performance of the Training/Validation pairwise model is evaluated using only the query compound information. The training compound is replaced with a random vector of the same size that is sampled from a bit-wise Bernoulli distribution. The parameters of the Bernoulli distribution

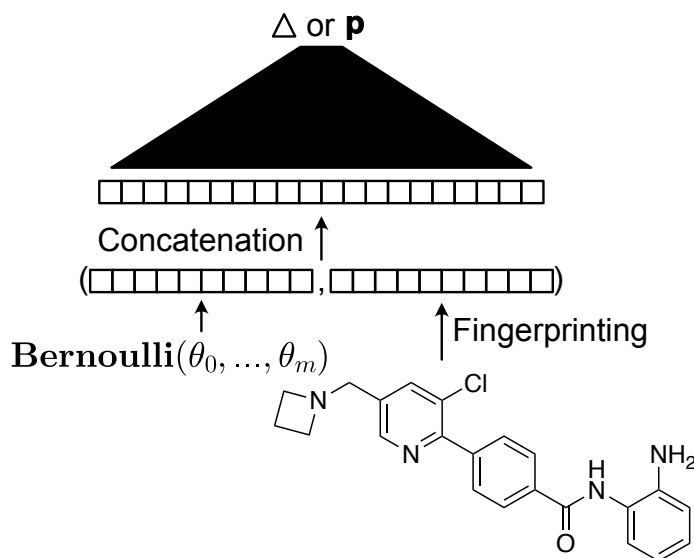


Fig. 4.8. Architecture of query-only predictions. Training compounds are replaced with random vectors sampled from a bit-wise Bernoulli distribution according to average bit densities across the dataset.

are the bit-wise average density of the fingerprints. This model is shown graphically in figure 4.8. In this case, the model is still trained on complete pairs and is simply validated on pairs with the training compound replaced in this way.

The results of the query-only experiments are shown in figure 4.9, with density colouring added because the outline of the scattered points alone does not adequately show the significant performance difference. The AurA dataset and the ECFP fingerprint type are used for the example. We can see that the performance drops off when the model loses access to the true training compounds. This seems to indicate that both the training and validation compounds are important to prediction. This may indicate whether the model is learning some kind of pairwise interactions between training and query compounds, however it does not prove this. Under the described conditions, these models could still be learning to predict each compound's activity independently and learning to internally compute the difference. When the training compound is replaced by a random vector, the prediction for that half of the input would become worse, making the overall difference prediction worse. Another way to possibly test for learning pairwise interactions would be to do an ablation study masking out various sets of input bits that span both input compounds, looking for combinations of bits or features that indicate "high difference" in activity when present together, but do not indicate either high or low activity when present alone.

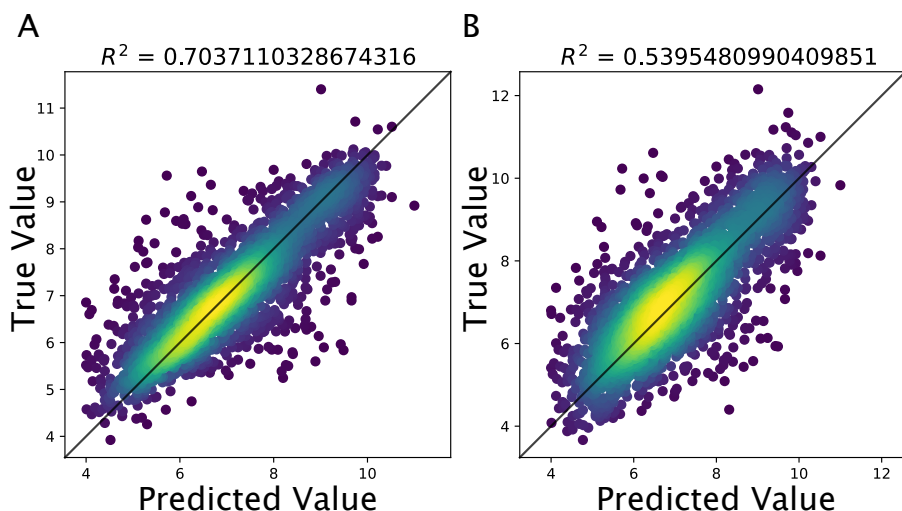


Fig. 4.9. Scatter plots showing A) The performance of a trained model on complete Training/Validation pairs and B) The performance of a trained model on query-only pairs with random vectors as training compounds. These results are for the AurA dataset and ECFP fingerprint type, with a trained predictor network architecture optimized by HPO.

We can see that there is some general prediction performance provided by the query compound alone, but a more refined prediction is possible when training compounds are used. This process was repeated for each of the three datasets and for the two algorithmic fingerprints, ECFP and RCFP. The results of these experiments are shown in figure 4.10. The GCN fingerprints were excluded because their final fingerprints are not binary bit vectors like ECFP and RCFP, so the Bernoulli sampling is not possible.

We can see from these results that across both fingerprints and all datasets, the prediction performance on full pairs is much better. The performance difference varies between the datasets, which could be due to several things. It could possibly indicate which datasets need predictors that recognize pairwise information and which do not. For example, performance of ECFP on the CEP dataset suffers significantly when the query-only situation is applied, which may indicate how important pairs are to that specific model. Alternatively, if the model is only predicting independent properties and computing their differences, the performance difference between datasets could be related to the average property difference between pairs of compounds in each of the datasets. Consider if one compound in a pair is replaced with a random vector, then the difference prediction would be based on the remaining compound alone and the accuracy of that prediction may depend on the average pairwise property difference in the dataset.

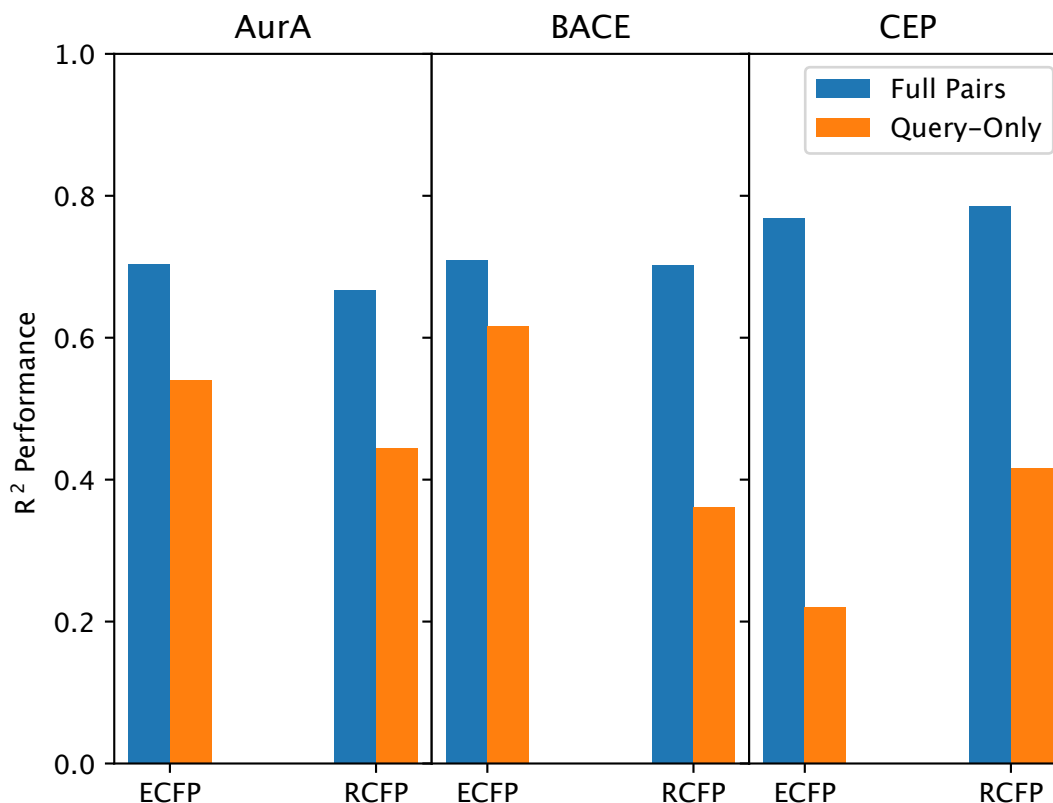


Fig. 4.10. R^2 performance of various pairwise models trained on random pairs from various dataset in the standard "full pair" validation situations and also in the query-only situation.

4.4. Validation-Validation (VV) Pairs

For the VV validation type, both compounds are validation compounds and have unknown activity. The output of the predictor is then difference in activity, and unlike the TV validation type, the individual activities of either compound cannot be isolated from this prediction. There is also no averaging component or predictions distributions as in the TV validation case. The predictor structure is shown in the top left panel of figure 4.3.

To compare the single prediction approach to pairwise VV, pair predictions are sampled from the single predictions. Single predictions are made for all validation compounds and from these, pairs are sampled to obtain pairwise predicted values of the same form as the pairwise VV predictions, this process is shown in figure 4.11.

4.4.1. Performance Compared to Single Prediction

The performance of the pairwise VV approach is compared to the single approach, with results shown in figure 4.12. Performance is presented in R^2 for the three datasets and all four

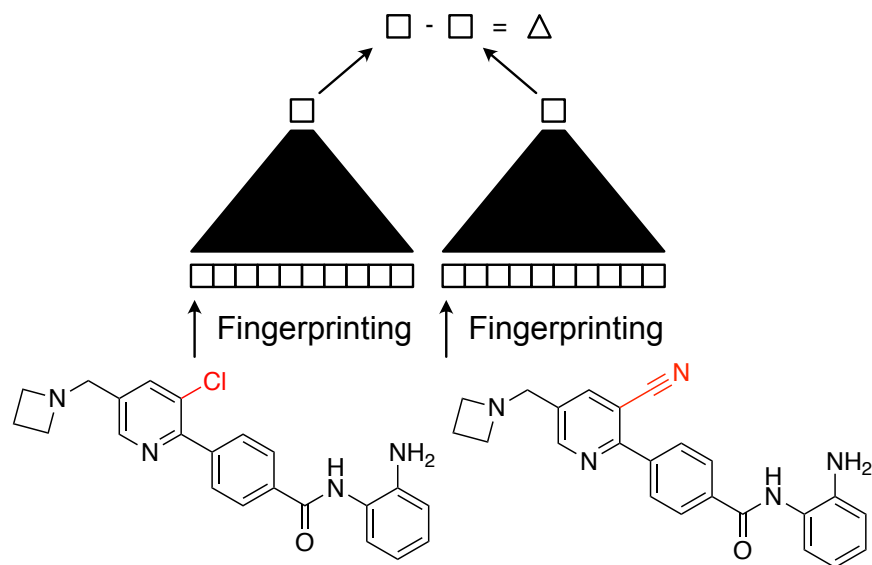


Fig. 4.11. Architecture of the single approach with pairwise sampling, which serves as the comparison baseline for the VV validation type. The predictors on the left and right are the same single predictor, which was trained on single properties.

of the discussed fingerprints. Each bar represents an individual hyperparameter optimization for that fingerprint type, dataset and approach (pairwise vs single) as described in section 2.5. Each HPO optimization step was conducted for 15,000 training gradient steps for the single approach and 25,000 steps for the pairwise VV approach.

Similar to the TV case, there appears to be no consistent performance improvement from using the pairwise approach over the single approach. A possible reason for this could be that the pairwise MLP network has to memorize so much more than the single approach. In the single approach with pairwise sampling, the network and input are half the size, and the scope of possible recognizable features only spans the compound space and not the entire pair space.

The single approach may perform so well because the subtraction operation needed to turn single predictions into pairwise ones may be a good approximation of how the MLP learns to treat different compounds. One could imagine the model learning a latent activity for each compound then calculating the difference internally. This would be advantageous because each pair occurs infrequently, but the constituent compounds occur much more frequently so having a go-to latent activity prediction for each one may be how the model learns to do this.

In addition to this, the single approach with pairwise sampling shown in figure 4.11 is a tied structure where each half of the input is dealt with in an identical way. In the

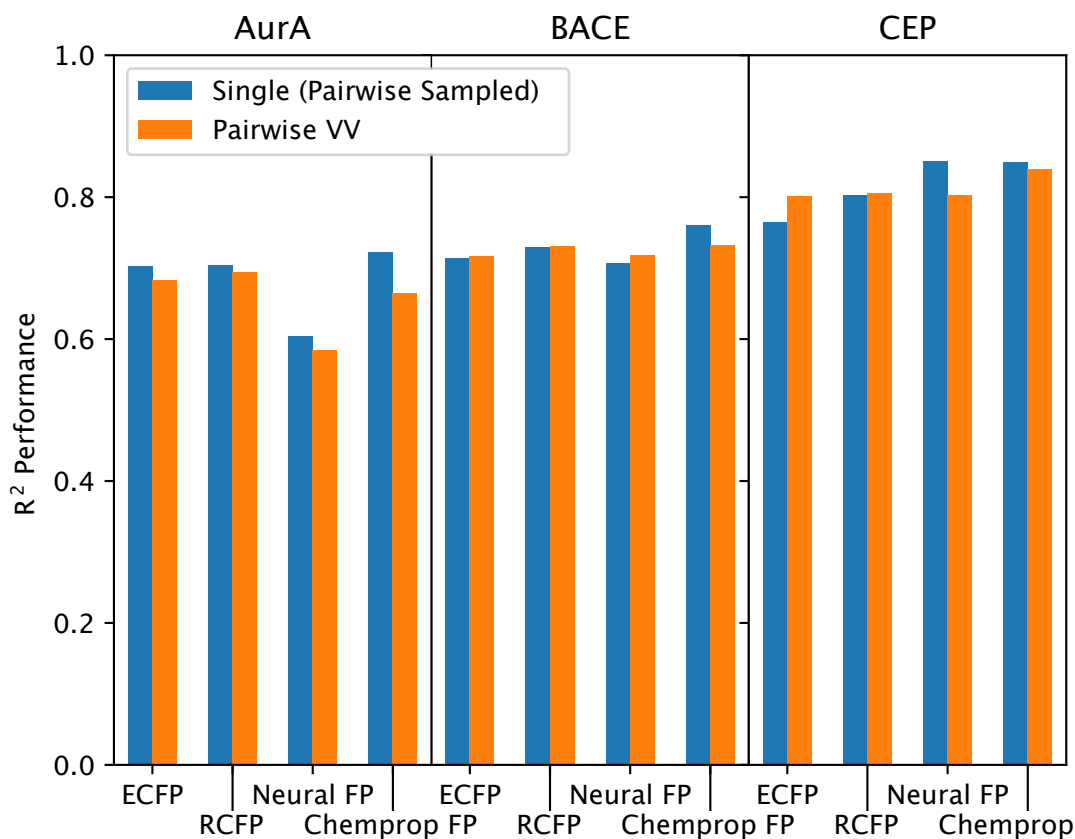


Fig. 4.12. The performance as measured by R^2 on validation/validation pairs for several fingerprint types and datasets. Compared with the performance obtained from pairwise sampling predicted values from the corresponding single approach.

pairwise VV approach, each half of the input is treated differently, which may be detrimental. For example, the single approach with pairwise sampling ensures the mean of all predicted property differences is centered at 0. This is the reality but it is not guaranteed in the pairwise VV case, perhaps if this was enforced in the predictor it could improve performance.

Property difference predictions on their own have limited utility in medicinal chemistry, as most of the time pairs are examined, one of the two activities is known. This is especially true when methods like MMPA are used to suggest new compounds or suggest the activity of a compound with unknown activity. A possible use of property differences alone could be to compile a more intelligent pair rule database that does not rely on strict substructure definitions.

4.5. Training/Validation Split Types

Up to this point we have only considered randomly splitting compounds into the training and validation sets, as was discussed in section 1.3.1. Common in many machine learning fields, with random split, the dataset is randomly shuffled and split into equal portions according to the number of training folds. In this work, 5-fold cross validation split was used.

Although this is the more common approach for general machine learning, there is evidence that it is a poor choice for QSAR applications [75, 92]. The arguments against random splitting for QSAR are based on how predictive models are applied in medicinal chemistry projects. In the iterative drug development process described in section 1.4.1, particularly in hit-to-lead development and lead optimization, new compounds are developed iteratively based on structural changes to previous compounds. This means that a compound is likely to be more structurally similar to compounds that were synthesized around the same time as it. This is the motivation behind a time-based training/validation split.

In time-based training/validation split, QSAR models are trained on structures with known properties, that were synthesized and tested in the past. The models are then used to predict the unknown properties of structures that will be synthesized and tested in the future. There is a tendency for those unknown structures to be the result of pushing a medicinal chemistry investigation into new structural areas, meaning that a time-split approach results in a different structural populations of compounds between training and validation. This means that the exact task of prediction under time-split conditions is a soft transfer-learning problem where a small difference between training and validation data distributions is expected.

4.5.1. Scaffold Splitting

As mentioned above, a hypothetical gold-standard for training/validation split for QSAR tasks would be a time split. However this is not practical due to datasets often not having timestamps for the testing of various properties and if they did, this method would still remain unsuitable for multi-fold cross validation and would only give one possible validation fold (the most recently tested compounds).

Sheridan 2013 [75] proposes a surrogate split type meant to approximate a time-split to provide a better real-world evaluation of model performance. Since the tendency for medicinal chemistry programs is for the structure to change and progress over time, there are natural divisions within the dataset based on structural grounds. Dividing the dataset

along these structural lines can approximate different eras of the investigation and provide a split type that is similar to time-split, referred to as scaffold split.

The scaffold split performed in this work follows the implementation of scaffold splitting in the Deepchem/Moleculenet software package [93]. The datasets were first divided into "scaffold groups" according to their scaffolds determined by the Bemis-Murcko scaffolding algorithm [6], then the resulting clusters are randomly sorted into the different validation folds. During this random sorting process, one scaffold group was added to each validation fold before adding to a second to any of the folds, to keep the assignment process random but also balanced in size.

4.5.2. Performance Comparison Between Split Types

Figures 4.13 to 4.15 show the performance measured by R^2 on the different datasets split randomly and split according to their scaffolds, also under pairwise and single approaches. Each bar represents an individual hyperparameter optimization for that fingerprint type, dataset and approach (pairwise vs single) as described in section 2.5. Each HPO optimization step was conducted for 15,000 training gradient steps for the single approach and 25,000 steps for the pairwise TV approach.

We can see that random split results are better than scaffold splitting for most datasets and fingerprints. This is expected, as prediction under scaffold split conditions is fundamentally a more difficult task. The performance difference between random and scaffold is biggest in the AurA dataset and relatively mild in the BACE and CEP datasets. Scaffold splitting even has a higher performance compared to random split for the single approach with ECFP on the CEP dataset.

The higher relative performance of scaffold splitting on certain datasets could be due to the structure of the dataset. If most compounds in the dataset are not very similar to each other, there will not be many compounds sharing the same scaffolds and therefore the "scaffold groups" could contain as few as one compound. In this case, scaffold split is essentially just random split.

The performance difference between scaffold split and random split is roughly the same for both the pairwise and single approaches, meaning that the additional difficulty of the scaffold splitting task is consistent across the two approaches. In addition, the pairwise approach also does not appear to add any performance to the scaffold splitting scenario, when compared to pairwise TV, random split.

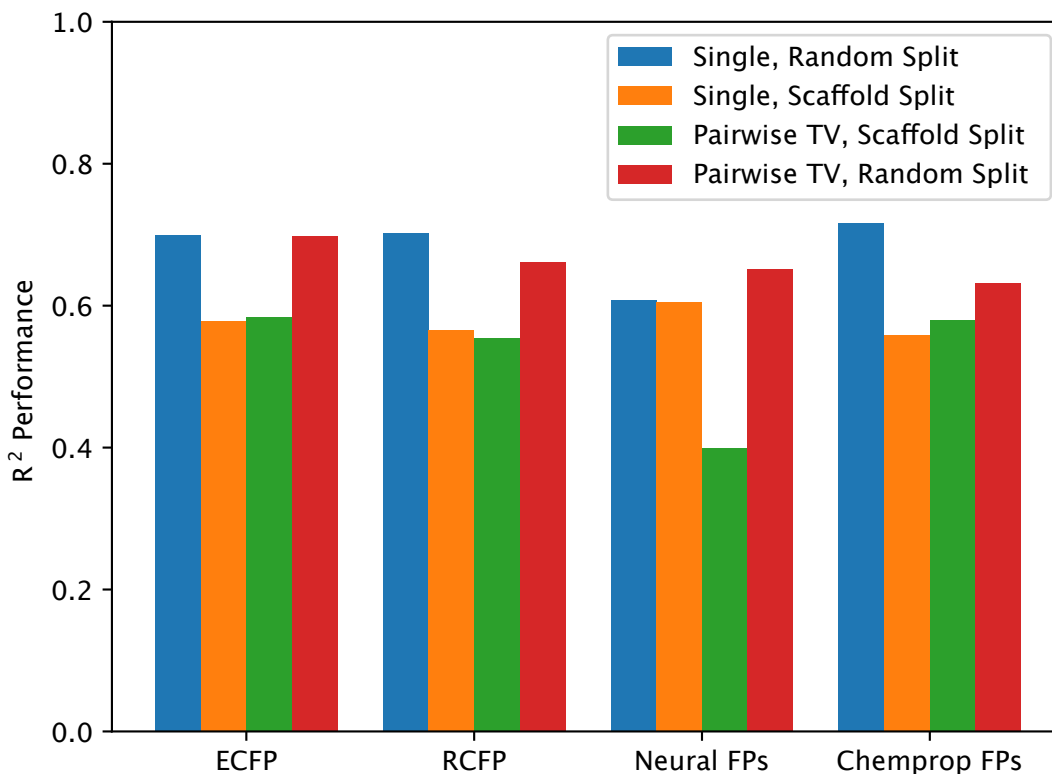


Fig. 4.13. The performance as measured by R^2 on training/validation pairs compared between random and scaffold dataset split for several fingerprint types, on the AurA dataset.

4.6. Pair Types

The possible pair space, as mentioned above, is $\mathbf{N} \times \mathbf{N}$ if self-similar pairs are included or $\mathbf{N} \times (\mathbf{N} - 1)$ if they are not. When sampling pairs randomly from this space, there is no guarantee that two compounds will form a meaningful pair. A "meaningful pair" is loosely defined, but we can consider it meaning that there is enough similarity between the compounds for the predictor to learn something about the pair as a whole. For example, if there is absolutely no similarity at all between two compounds, we can imagine the predictor must predict each compound's activity independently, then subtract them to predict a difference. This would be the predictor doing single-compound predictions, and any potentially useful pairwise information is unused.

Predictions in this section are the TV validation type, where multiple predictions are made for each validation compound and the results are averaged to make the final prediction, as described in section 4.3. In the random pairs case, a fixed number of compounds are randomly sampled to make a query compound's prediction distribution. For the following

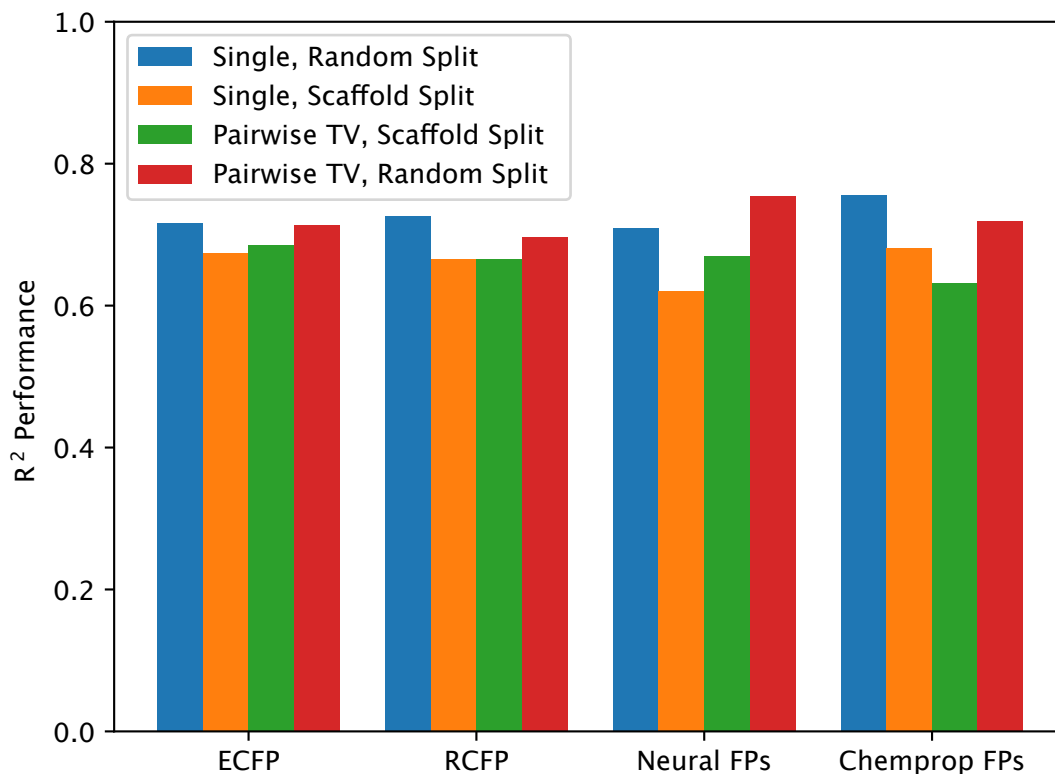


Fig. 4.14. The performance as measured by R^2 on training/validation pairs compared between random and scaffold dataset split for several fingerprint types, on the BACE dataset.

strict pair types, the prediction distributions come from all of the pairs allowed for a query compound, which can range from 1 pair to over 100 pairs.

4.6.1. Different Pair Type Descriptions

The three different pair types/sources are Random Pairs, MMPA pairs and Threshold Pairs. Each pair type has different requirements that are outlined below. In addition to these definitions, in order for a pair to be valid its members must still belong to the proper training/validation set. For example, TV pairs must feature one training and one validation compound as well as obeying the following requirements for the strict pair types.

4.6.1.1. Random Pairs. For this type, a pair is simply a pairing of any two compounds in the data set. These are the type of pairs used in all of the above experiments. This results in $N \times N$ possible pairs where N is the number of compounds in the dataset. The pairs are sampled from all possible pairs as the mini-batches are being made.

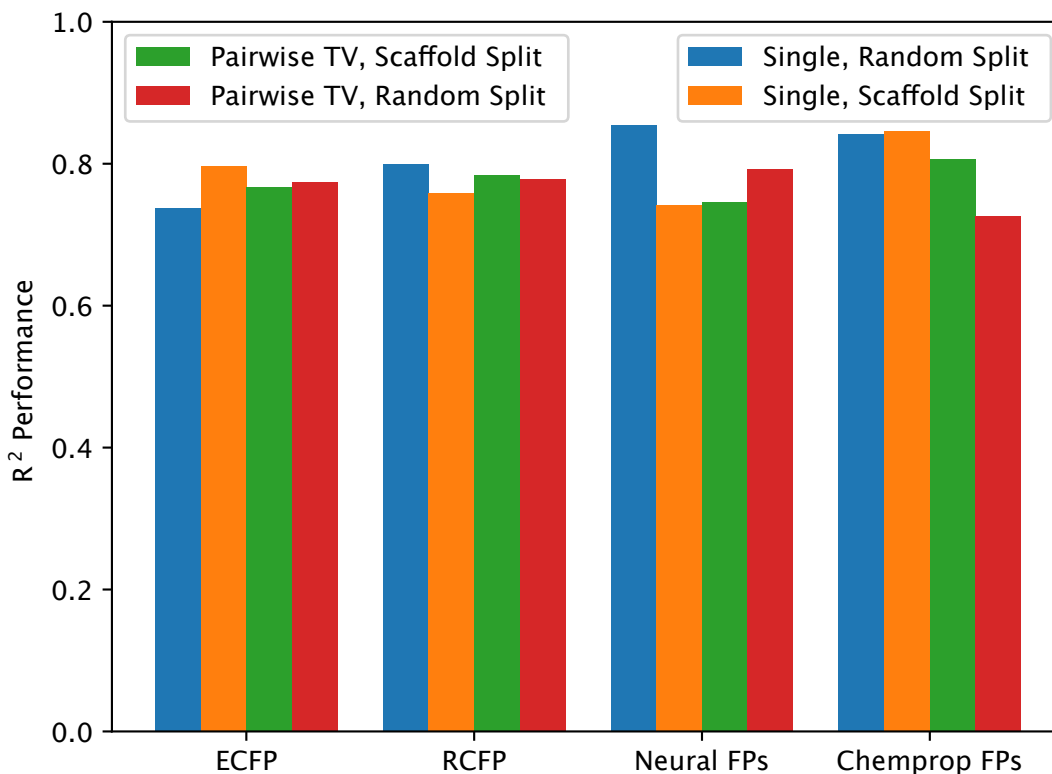


Fig. 4.15. The performance as measured by R^2 on training/validation pairs compared between random and scaffold dataset split for several fingerprint types, on the CEP dataset.

4.6.1.2. Matched Molecular Pairs. This pair type ensures that the pairs of compounds are sufficiently similar so as to represent a meaningful chemical difference. The definition of "meaningful chemical difference" is certainly subjective, but substructure overlap is a possible interpretation. With overlapping sub-structures, we know that in a biological setting at least portions of the molecule may interact with their environment in the same way for both compounds. The pairs are generated using the method outlined in Hussain et al. 2010 [43] and implemented in RDKit. This is an example of a fragment-indexing method of MMP generation and scales well to large datasets.

This is similar to the pair generation approach found in Turk et al. [73]. In that work, MMP's are generated according to the BRICS decomposition algorithm, which are used as the pair source.

4.6.1.3. Similarity Threshold. This approach is similar in principle to the random pairs described above, but pairs are not used if their similarity does not meet a threshold. The chosen threshold for this whole work was 0.6 in Tanimoto fingerprint similarity for 2048 bit ECFP with radius 3. Pairs satisfying this requirement are a very small portion of all

possible pairs. In practice, compiling a list of the pairs ahead of time is more efficient than random sampling and discarding, due to the low proportion of pairs that actually satisfy the requirement. The algorithm for finding all the pairs of a given similarity is $O(N^2)$ where N is the number of compounds, so it is unsuitable for very large datasets. The pair enumeration for the CEP dataset took upwards of 10 hours to complete, compared to minutes for the smaller AurA and BACE datasets.

4.6.2. Comparison of Pair Types

Figure 4.16 shows the similarity distribution of the three pair types for the AurA dataset. Similarity in this case is defined as Tanimoto distance between ECFP of size 2048 and radius 3. ECFP is used instead of RCFP in this demonstration because ECFP similarity is a standard measure of chemical similarity in the literature. Random pairs are the most numerous (N^2 total pairs), so a random subset of them were sampled for the histogram. We can see that the majority of random pairs share very little similarity, with most pairs having less than 0.2 in Tanimoto similarity. MMPA and threshold pairs have significant populations in much higher areas of the similarity spectrum, both having their highest populations around 0.6 in Tanimoto similarity. Several compounds show 100% similarity, this is likely due to pairs of stereoisomers in the dataset, which are different molecules possessing the same connectivity and therefore fingerprint.

Figure 4.17 expands on the idea of figure 4.16 by scattering the pairwise similarity against property difference for the same pairs. For random pairs, there is again a high population in the low similarity section of the chart. These dissimilar pairs appear to often also have both high property and low property differences.

Both MMPA and threshold pairs are concentrated in the higher similarity region. There is a larger population in the high similarity and high property difference region of the chart, which is indicative of activity cliffs. If a predictor is exposed to more activity cliffs, this may result in understanding what makes an activity cliff for a given dataset. Activity cliffs are an example of pairs sharing "meaningful chemical information" because high similarity and high property difference mean that the small structural differences may be very important to the property of the compounds.

Figure 4.18 shows the distribution of how many pairs each compound makes with other compounds in the dataset, according to pair type. Random pairs are excluded from this graph because the number of pairs is $N - 1$ for each of the compounds. For MMPA and threshold pairs, the more restrictive pair requirements mean that most compounds actually have fewer than 20 pairs in the dataset, with a large number of compounds actually having no

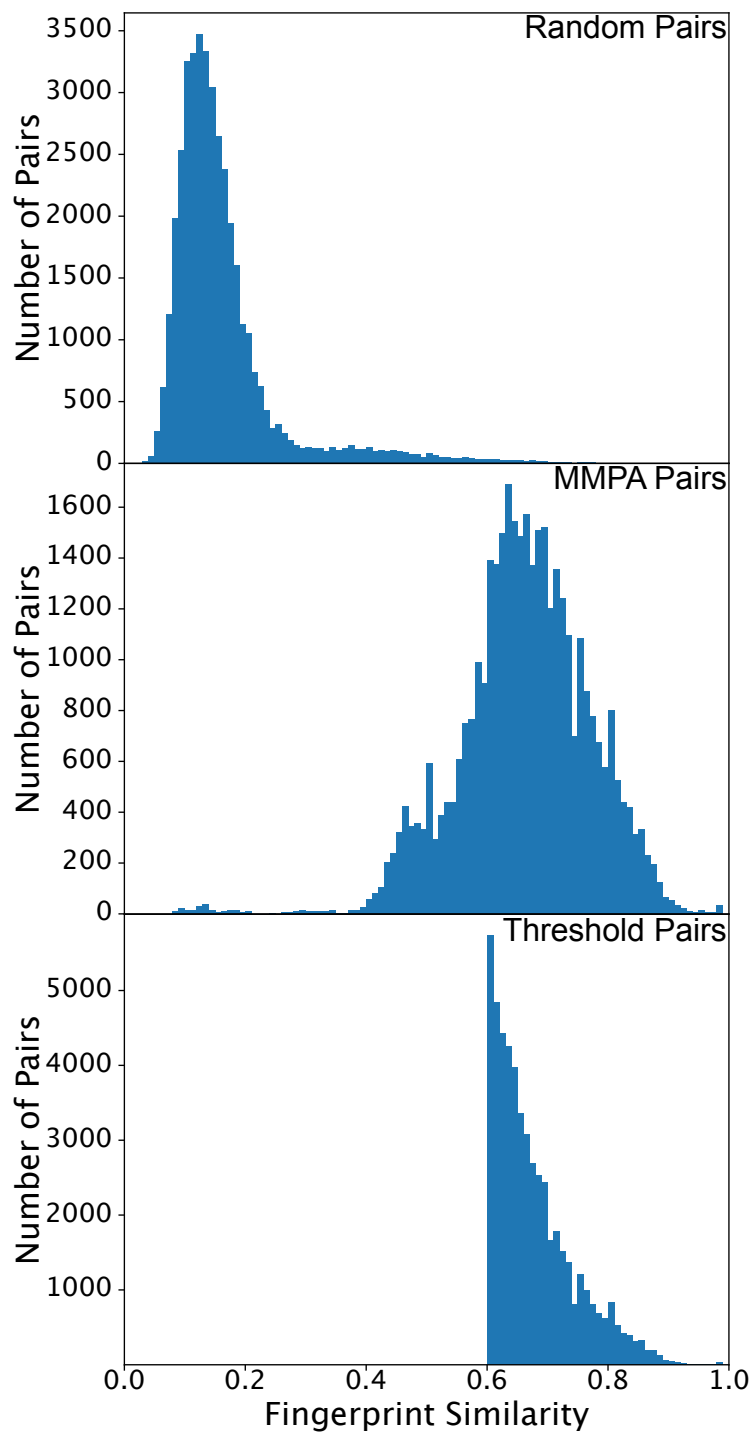


Fig. 4.16. Pairwise similarity distributions of the different pair types for the AurA dataset. A sampled subset is used for random pairs, whereas MMPA and Threshold use all available pairs of the given type. Similarity is Tanimoto distance between ECFP of size 2048 and radius 3. Chosen threshold was 0.6.

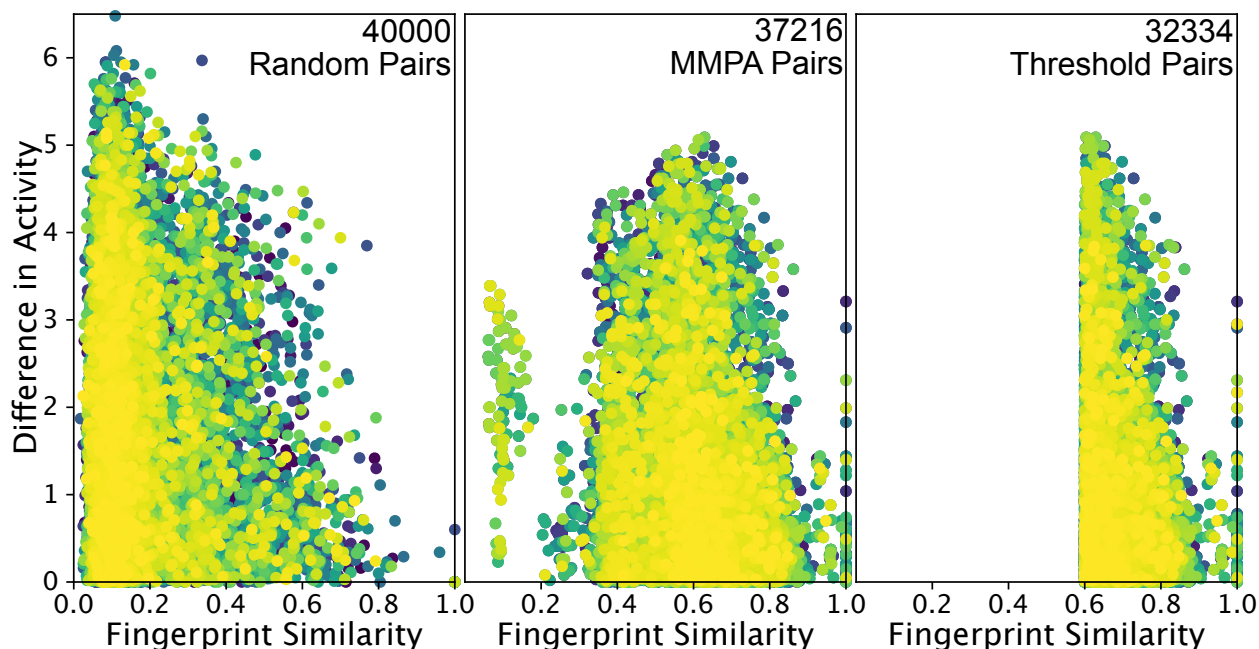


Fig. 4.17. Activity difference plotted against pairwise similarity, coloured by point density. Similarity is Tanimoto distance between ECFP of size 2048 and radius 3. Chosen threshold was 0.6.

pairs. This complicates the prediction even further because a pairwise predictor cannot use compounds without any valid pairs, so all of those compounds are left out of the prediction.

The number of useful pairs is reduced even further because even though a compound may form a pair with another compound somewhere in the dataset, depending on the validation type (TV or VV) and the training/validation split, that pair may not be eligible. For example in the TV case, if a query compound has only one pair, which happens to also be in the validation set, then no training/validation pairs exist for that query compound.

4.6.3. Performance Comparison Between Pair Types

Figure 4.19 shows the performance measured by R^2 on the different datasets and fingerprints using the different pair types. Each bar represents an individual hyperparameter optimization for that fingerprint type, dataset and pair type as described in section 2.5. Each hyperparameter optimization step was conducted for 25,000 gradient steps.

We can see from these efforts that for the most part, random pairs appear to perform better as a training dataset than the so-called "curated" datasets of MMPA or threshold pairs do. This could be because the random pairs, although they contain many dissimilar pairs, also contain the whole host of MMPA and threshold pairs. In the MMPA and threshold pair

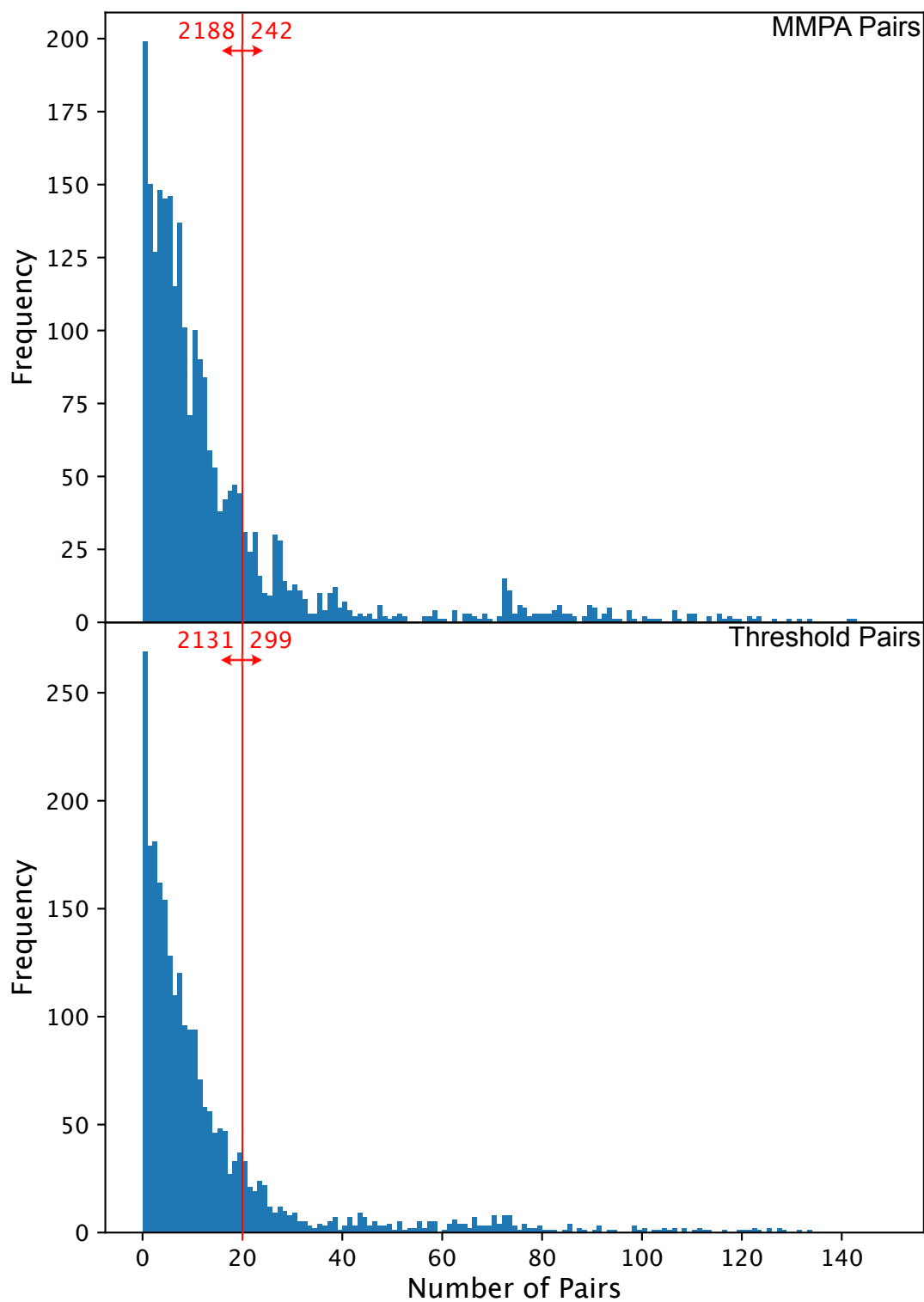


Fig. 4.18. Distribution of the number of MMPA and Threshold pairs each compound has in the dataset.

types, the smaller datasets of pairs may be too restrictive in the diversity of compounds and

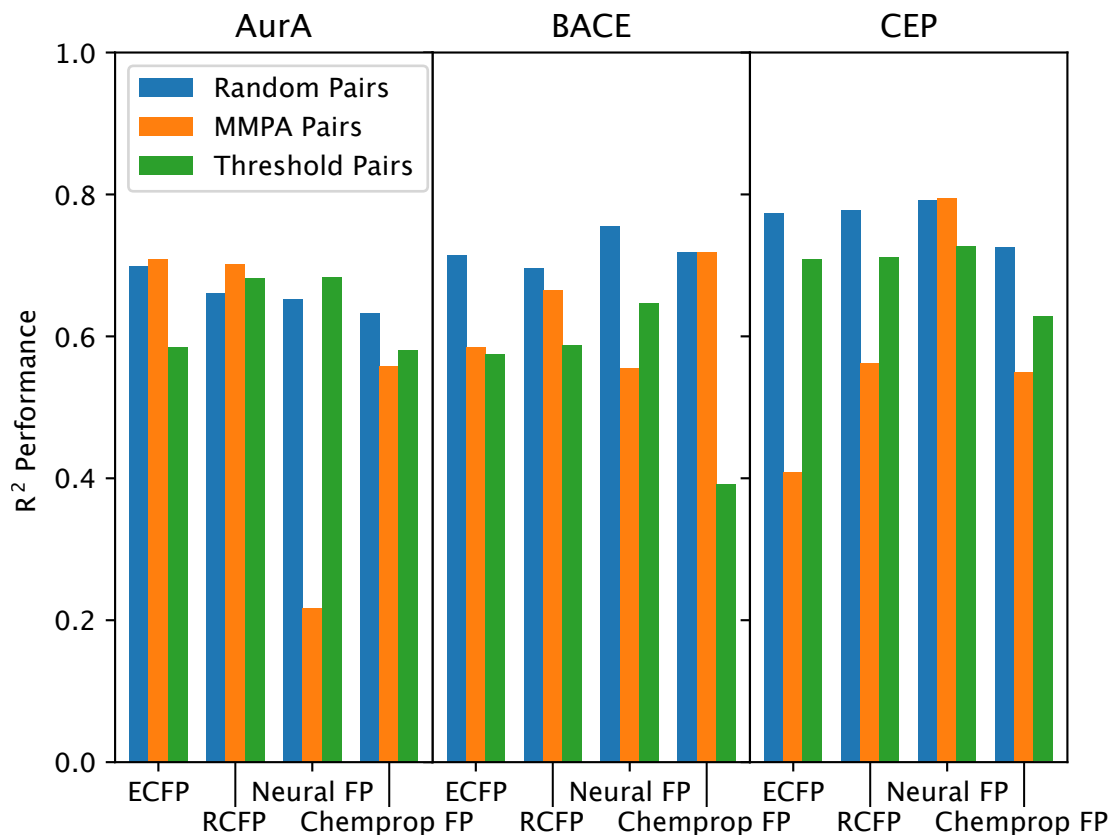


Fig. 4.19. The performance as measured by R^2 on training/validation pairs compared between random, MMPA and threshold pairs for several fingerprint types and datasets.

pairs that the network suffers from this. The curated pair types are also trained sequentially through the same random order of possible pairs over and over again instead of randomly sampling across a large spectrum.

The random pair source has access to all of the other pairs from both of the other pair types, they are just surrounded by an ocean of other less similar pairs. Training on these less similar random pairs may not be detrimental to the performance of the method considering the very long training timescales that were used where all pairs, similar or dissimilar, were seen by the model.

Experimenting with different thresholds may yield more interesting results. The threshold of 0.6 was chosen so the number of threshold pairs would roughly match the number of total MMPA pairs for the AurA dataset. Maybe if the threshold was lowered to allow for substantially more pairs, the model would train as well as it does on random pairs and still have a bias in the dataset towards more similar pairs. Perhaps a compromise would be to

| Pair Type | Split Type | TV pairs | VV pairs |
|---------------|------------|----------|----------|
| Random | Random | 944784 | 236196 |
| Random | Scaffold | 944784 | 236196 |
| MMPA | Random | 5558 | 1276 |
| MMPA | Scaffold | 4254 | 4676 |
| Threshold 0.6 | Random | 3628 | 880 |
| Threshold 0.6 | Scaffold | 3375 | 720 |

Table 4.1. Number of pairs of each validation type under the different pair types and validation splits, for the AurA dataset with 0.2/0.8 valid compound/train compound split.

train on the most similar 50% of all possible pairs which from the distribution in figure 4.16 would be around 0.1.

4.6.4. Pair Types with Scaffold Split

Notably absent from the above results is the scenario where we combine a non-random pair type with a scaffold dataset split. This was excluded because the pair type results reveal that the random pairs perform just as well or better than the MMPA or threshold pairs. However, using MMPA or threshold pairs with a scaffold dataset split does introduce an interesting situation that we will discuss in this section.

With a scaffold split, both the training and validation sets are composed of multiple groups of compounds, each with a common scaffold. On top of this, MMPA or threshold pairs will restrict the allowed pairs to only compounds that are similar. In this situation, we are far more likely to find allowed pairs for a compound within the same validation fold as that compound since that is where its scaffold group is. This means a higher population of VV pairs over TV pairs, and brings into question the validity of the TV validation type in this situation.

More fundamentally, the TV validation type requires that allowed pairs exist between training and validation compounds. We would then expect them to be similar if those pairs are MMPA or scaffold pairs. For the scaffold split, we would expect the training and validation sets of compounds to be structurally different. These are conflicting expectations and shed doubt on the combination of these two approaches. See table 4.1 for the number of pairs of various types under the different training/validation splits for the AurA dataset.

There is a case to be made for a combined training/validation split and pair type generation algorithm to simultaneously find pairs and divide the dataset into training and validation. This way the conflicting expectation mentioned above could be avoided. This is similar to the approach taken in Turk et al 2017 [73]. In that work, the authors consider

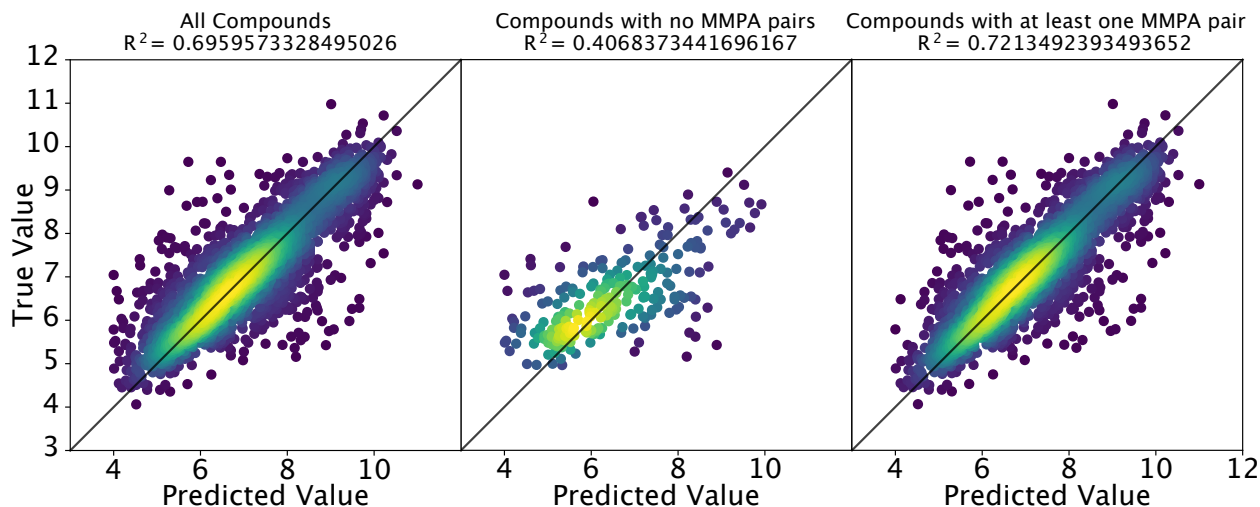


Fig. 4.20. Comparison of prediction scatter plots for datasets masked by how many pairs each compound has. Predictions are from a pairwise TV model trained on random pairs and ECFP fingerprints.

two training/validation "tasks" where the pair types are created alongside the splitting of the dataset. Both tasks are specific cases of Validation/Validation pairs, where the pairs are selected based on very specific structural grounds (using 3D protein structure). A combined method was not explored in our work, to keep the investigation as comparable to the single approach as possible by first performing the training/validation split separately from curating dataset of pairs.

4.7. Pairwise Dataset Masking

Although the pairwise model appears to have worse performance on the curated pair datasets, we can use the MMPs in these datasets to improve the prediction performance of other models. By counting the number of MMPA pairs each compound has in the dataset, and masking the dataset based on those numbers, we can see a relationship between the number of pairs a compound has and the prediction performance of a model on that compound. Figure 4.20 shows an example of this relationship. The predictions from a TV pairwise model trained on random pairs were collected. Then the performance of these predictions was evaluated, after masking the dataset to remove compounds that do not have MMPs in the dataset.

We can see in the figure 4.20 that the overall performance on the predictions is improved when the compounds with no MMPA pairs are removed from the calculation, comparing the panel on the far right to the one on the far left. The middle panel shows the performance on

only the compounds with no MMP's, which is actually low in comparison to those compounds with MMP's. This result makes sense in the context of a learning problem, ML systems tend to perform better when they are tested on data that is similar to what they have seen in training. When a compound has MMPs, it has guaranteed structural overlap with some other compounds in the dataset, which means that the model has seen parts of that structure before, and we see this results in higher performance of the model.

Expanding on this idea, we make the masking threshold (how many pairs are needed for that compound to be included) a sliding one and plot the resulting performance R^2 value in figure 4.21. In this figure, the TV pairwise model trained on random pairs is compared to the single model, as well as masking based on either threshold pairs or MMPA pairs. The top row of plots in the figure are the performances when only compounds with at least a number of pairs are included. The bottom row are the performances when only compounds with exactly a number of pairs are included.

For example, the top left panel is the same predictor situation as figure 4.20. For $x=0$ of the top left panel of figure 4.21, we get the situation in the leftmost panel of figure 4.20 where all compounds are included for this predictor. Moving to $x=1$, we get the situation in the rightmost panel, where the performance is reported on all compounds with at least one pair. The middle panel of figure 4.20 is reflected in the bottom left panel of 4.21 when $x=0$. At this point, the compounds included in the performance calculation are only those with exactly zero MMPs.

Figure 4.21 shows us the same effect as seen in 4.20, which is that predictors tend to perform worse when predicting the properties of compounds that have no MMP pairs in the dataset. We can see that the same is true for threshold pairs as well, with the compounds with no pairs.

This result is particularly interesting because the pairwise model has no knowledge of these curated pairs and despite this, the number of MMPA pairs is still a telltale sign of predictive performance. Going beyond this, even the single compound approach also shows the same effect. The single approach has no knowledge of pairwise information at all, let alone being trained on specific kinds of pairs like the MMPA models.

This suggests that what's shown here is more a property of the dataset and how similarity in structure (and therefore number of pairs) relates to similar property values. This is a central axiom of SAR and molecular biology, which is that similar compounds result in similar activities. Specifically in these results, we see this notion alongside the notion of generalization in neural networks, how predictors perform better on data they have seen before.

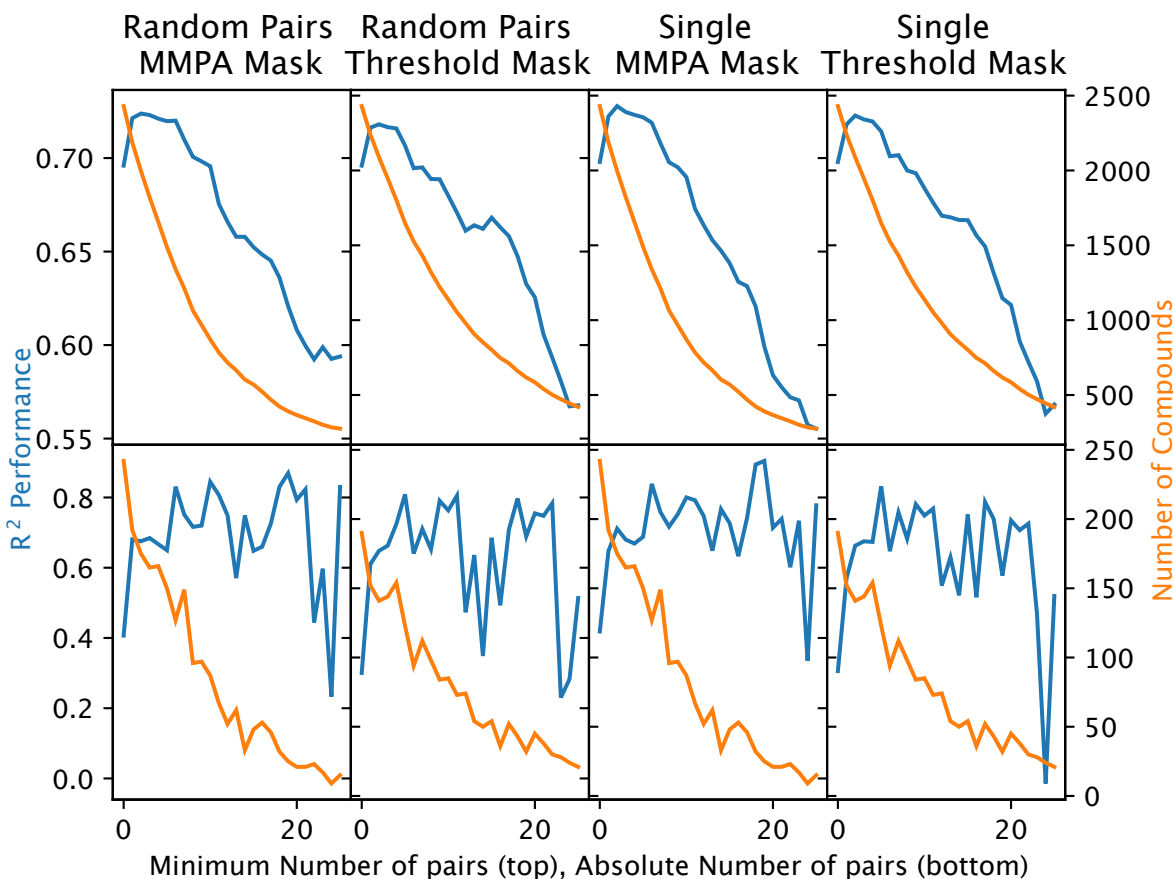


Fig. 4.21. Prediction performance as a function of number of either MMPA pairs or Threshold pairs that each compound has in the AurA dataset. The dataset is masked for each performance value, evaluating performance only on pairs with either at least (top) or exactly (bottom) a certain number of pairs in the training set.

From this investigation, we chose to use MMPA pairs over threshold pairs for further investigations because MMPs are more common in practice for medicinal chemistry efforts and are more familiar to chemists. For the predictor, we chose the single predictor because it is more common and simpler. We see the same performance between both the single and pairwise so we chose the more common one. Figure 4.22 explores similar results for the single predictor using MMPA pair masking but for the other datasets.

The results for AurA and BACE are quite similar, observing the same effect as seen on AurA. The results for CEP are unlike the other two. First off, this dataset is much larger than the other two. In addition to this, it also appears to have a very high diversity, shown by the fact that there are no compounds that have more than 8 matched molecular pairs in the entirety of the 30,000 compounds dataset. A similar effect is still observed where more

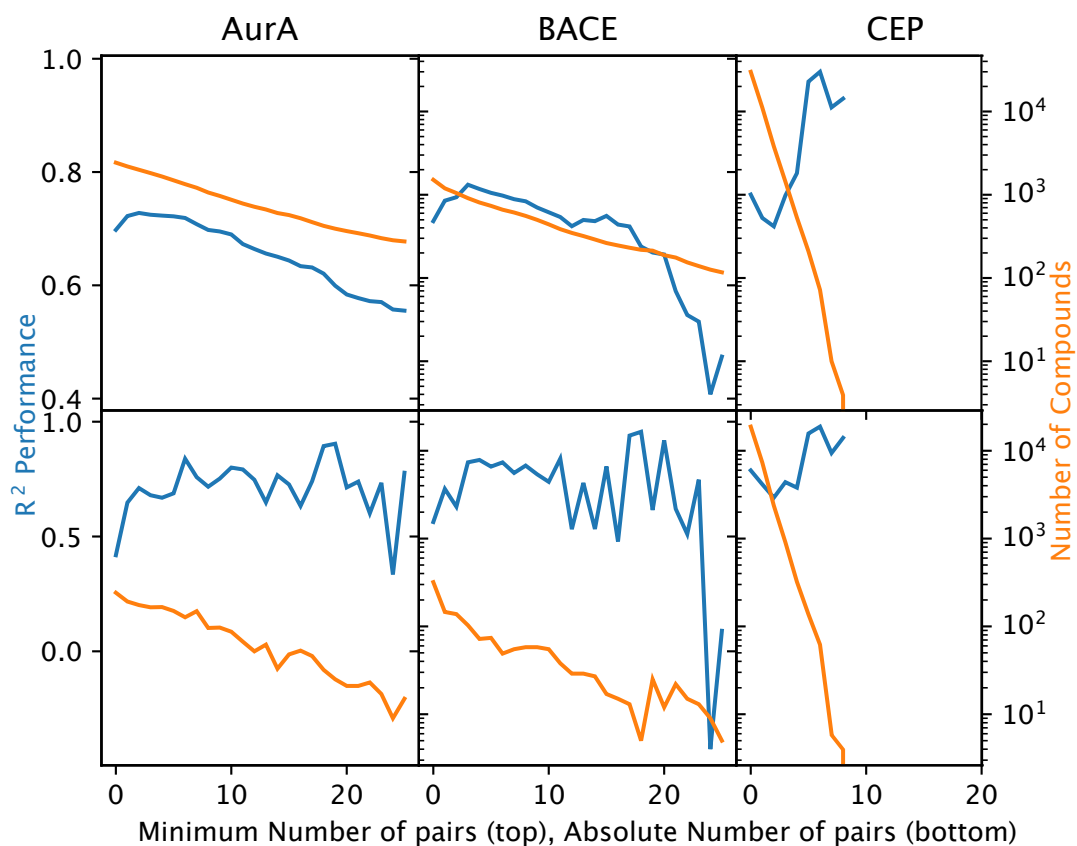


Fig. 4.22. Prediction performance of a single predictor as a function of number of MMPA pairs each compound has in each of the datasets. The datasets are masked for each performance value, evaluating performance only on pairs with either at least (top) or exactly (bottom) a certain number of pairs in the training set.

MMPs means higher predictive performance, with a peak in predictive performance when compounds that have 6 or 7 MMPs.

In all of these results, there appears to be a drop in performance at a high number of MMPA pairs. This may indicate that for compounds with exceptionally many MMPA pairs, those compounds may be significantly different than the rest, e.g. much smaller, which may negatively affect prediction performance.

The results from these database masking experiments have interesting implications for predictive QSAR models. For example, this can be used as a rough measurement of prediction uncertainty. Simply count the number of pairs that a compound has within the training set and understand that the predictive model will likely have a more difficult time predicting the properties of compounds with no MMPs.

These results can also provide motivations in library design and selection. If a library is being created to train a model for predictions, there appears to be an advantage to ensuring that MMPs exist within the dataset and between the predicted compounds and the dataset. This suggestion forms an interesting problem when compared to the traditional notion that a chemical library for tasks such as screening should be exceedingly diverse so as to cover as much of the chemical space as possible [88]. Further investigations may find a balance between chemical space coverage and number of MMPs.

Chapter 5

Conclusion

Our goal was to propose more interesting, powerful and explainable representations of chemical compounds to improve predictive models. Predictive QSAR models are intrinsically tied to the practical medicinal chemistry projects they are used in, and we wanted to deepen this connection. Within predictive QSAR, we chose to direct our interest towards the hit-to-lead and lead development stages of the drug develop pipeline. These steps are marked by the complex multi-property optimization that is faced by medicinal chemists as they balance optimizing activity alongside pharmacokinetic properties, all under modest dataset sizes. We believe these development stages need more explainable methods in order to deeply impact the process.

Our investigations took two angles, first we examined a possible improvement to a molecular fingerprinting method that has an intrinsic source of confusion, which is collision-prone hashing. This was the done by RCFP which was investigated in section 3. Fingerprint collisions are a known problem with ECFP and intuitively, they seem to be a potential source of confusion for a predictor. Our results from RCFP show that bit collisions are not as confusing to an MLP predictor as may be expected. Although this fact may not be surprising to data scientists who understands the model capacity of an MLP and the associated predictive ability, it may come as a surprise to a chemist with little experience of machine learning. We also showed that when the capacity of the predictor is intentionally reduced to that of a linear model, using RCFP results in higher predictive performance than ECFP. This has applications in the explainability of the predictors, as RCFP with a linear predictor presents a much more direct substructure-to-prediction signal than using collision-prone fingerprints with high capacity but ambiguous models. This is because although a one-way hash function is used in RCFP, with the dataset available, the hash function can be reversed-engineered by process of elimination as discussed in section 3.3.3

Possible new avenues to push this work further may include a hybrid collision-free/collision-prone fingerprint where a portion of the fingerprint with the most common

bits is designated collision-free and another portion with less common bits designated collision-prone. Another interesting investigation would be to build the explainability system that can trace importance in a prediction model back to individual RCFP bits to identify the substructures that are important to prediction.

In our second investigation, we took a medicinal chemistry tool and applied machine learning to it. This was done in the pairwise comparison approach in section 4. Matched Molecular Pair Analysis is a useful tool for medicinal chemistry and intuitively allows a better grasp on the relationship between structure and properties. Our results with the pairwise comparison approach appear to show that these comparisons add nothing to the system’s prediction performance. This is another fact that may seem counter-intuitive to medicinal chemists working in this field. Although pairwise rules are incredibly helpful to human understanding, the MLP does not easily gain additional knowledge from it. We also observed that a model trained on enough random pairs outperforms models trained on pairs with guaranteed similarity, such as MMPA pairs. However if those MMPA pairs are found, we can use the number of pairs that each compound has a rough estimate of how performant the trained model will be on those compounds. In addition to this, we found that there are only marginal improvements to using GCN based fingerprints over algorithmic ones for the two smaller datasets that were examined (AurA and BACE). In the largest CEP dataset, GCN fingerprints performed better than the algorithmic ones, possibly due to more data to train on.

The pairwise efforts could be further developed by devising a more intelligent way to combine the representations of two compounds. Concatenation of fingerprints is the most simple, but perhaps if the compounds are more effectively compared and contrasted, we may see improvements in the pairwise performance. Maximum common substructure information would be interesting but likely too computationally costly to compute for every pair. Perhaps a short list of common structural features (similar to a fingerprint) could be used to represent substructures in common.

The pairwise comparison investigation could also benefit from neural network architectures that better account for the pairwise nature of the input, such as the Siamese neural network, which shares the weight parameters between two input streams. Using this kind of network alongside a difference-oriented input encoding to intelligently combine the encoded representations could allow for a more powerful combination of the pairs of compounds than simple concatenation. On the theme of more complicated representations, larger and more expressive neural networks could be used instead of relatively shallow DNNs. An example

of this could be an MPNN featuring a soft cross-attention mechanism, where information in one compound causes the network to attend to certain information in the other compound.

The pairwise dataset masking efforts of this work could also be used to motivate chemical library design. If a dataset is being compiled that will be used to train a predictor, our work suggests maximizing the number of MMPA pairs that each compound has. Such decisions would be a balancing act between choosing diverse compounds to cover a large chemical space and choosing less diverse compounds to ensure many MMPA pairs.

Both of our investigations attempt to incorporate some degree of explainability to predictive cheminformatic models. Reduced-Collision fingerprints do this through explainability of the model, in the form of fingerprint bits that directly represent substructures. Pairwise comparisons do this through explainability of the input data, by looking at chemically meaningful paired comparisons. Our hope with this work is that explainability will remain an important consideration in predictive QSAR models, which will help stop the fields of medicinal chemistry and predictive cheminformatics from diverging any further. As more complex ML and GCN methods and predictors are developed for cheminformatics, this distances the field from the vast expertise of existing medicinal chemistry platforms and personnel. Fostering communication between data scientists and chemists is critical for the symbiotic advancement of machine learning in medicinal chemistry. Both of the investigations of this work hope to be accessible to synthetic chemists who may question the utility of fingerprints that are prone to collisions, or may be familiar with matched molecular pair analysis. These points are part of a larger conversation of explainability in predictive cheminformatics that we hope to contribute to.

References

- [1] Dimitris K Agrafiotis, Maxim Shemanarev, Peter J Connolly, Michael Farnum, and Victor S Lobanov. SAR Maps: A New SAR Visualization Technique for Medicinal Chemists. *Journal of Medicinal Chemistry*, 50:5926–5937, 2007.
- [2] Tomoo Aoyama, Yuji Suzuki, and Hiroshi Ichikawa. Neural networks applied to pharmaceutical problems. III. Neural networks applied to quantitative structure-activity relationship (QSAR) analysis. *Journal of Medicinal Chemistry*, 33(9):2583–2590, September 1990.
- [3] Magdalena Bacilieri and Stefano Moro. Ligand-Based Drug Design Methodologies in Drug Discovery Process: An Overview. *Current Drug Discovery Technologies*, 3(3):155–165, September 2006.
- [4] Jürgen Bajorath. INTEGRATION OF VIRTUAL AND HIGH-THROUGHPUT SCREENING. *Nature Reviews Drug Discovery*, 1:882–894, 2002.
- [5] Richard Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–516, November 1954.
- [6] Guy W. Bemis and Mark A. Murcko. The properties of known drugs. 1. Molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996.
- [7] James Bergstra, Dan Yamins, and David D. Cox. Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms. In Stéfan van der Walt, Jarrod Millman, and Katy Huff, editors, *Proceedings of the 12th Python in Science Conference*, pages 13 – 19, 2013.
- [8] Leo Breiman. Random Forests. *Machine Learning*, 45:5–32, 2001.
- [9] A. Crum Brown and Thomas R. Fraser. On the Connection between Chemical Constitution and Physiological Action. *Transactions of the Royal Society of Edinburgh*, 25(1):151–203, 1868. Publisher: Royal Society of Edinburgh Scotland Foundation.
- [10] Craig L. Bruce, James L. Melville, Stephen D. Pickett, and Jonathan D. Hirst. Contemporary QSAR Classifiers Compared. *Journal of Chemical Information and Modeling*, 47(1):219–227, January 2007.
- [11] AD Buckingham, PW Fowler, and Jeremy M Hutson. Theoretical studies of van der Waals molecules and intermolecular forces. *Chemical Reviews*, 88(6):963–988, 1988. Publisher: ACS Publications.
- [12] Raymond E Carhart, Dennis H Smith, and R Venkataraghavan. Atom Pairs as Molecular Features in Structure-Activity Studies: Definition and Applications. *J. Chem. Inf. Comput. Sci.*, 25(2):64–73, 1985.
- [13] Adrià Cereto-Massagué, María José Ojeda, Cristina Valls, Miquel Mulero, Santiago Garcia-Vallvé, and Gerard Pujadas. Molecular fingerprint similarity search in virtual screening. *Methods*, 71:58–63, January 2015.
- [14] H.C. Stephen Chan, Hanbin Shan, Thamani Dahoun, Horst Vogel, and Shuguang Yuan. Advancing Drug Discovery via Artificial Intelligence. *Trends in Pharmacological Sciences*, 2019.

- [15] David E Clark. Virtual Screening: Is Bigger Always Better? Or Can Small Be Beautiful? *Journal of Chemical Information and Modeling*, 60(9):4120–4123, 2020.
- [16] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.
- [17] John G. Cumming, Andrew M. Davis, Sorel Muresan, Markus Haeberlein, and Hongming Chen. Chemical predictive modelling to improve compound quality. *Nature Reviews Drug Discovery*, 12(12):948–962, 2013.
- [18] Andrew Dalke, Jérôme Hert, and Christian Kramer. mmpdb: An Open-Source Matched Molecular Pair Platform for Large Multiproperty Data Sets. *Journal of Chemical Information and Modelling*, 58:902–910, 2018.
- [19] Laurianne David, Amol Thakkar, Rocío Mercado, and Ola Engkvist. Molecular representations in AI-driven drug discovery: a review and practical guide. *Journal of Cheminformatics*, 12(1):56, December 2020.
- [20] Andrew Davis, David Keeling, John Steele, Nicholas Tomkinson, and Alan Tinker. Components of Successful Lead Generation. *Current Topics in Medicinal Chemistry*, 5(4):421–439, May 2005.
- [21] Jörg Degen, Christof Wegscheid-Gerlach, Andrea Zaliani, and Matthias Rarey. On the art of compiling and using ‘drug-like’ chemical fragment spaces. *ChemMedChem*, 3(10):1503–1507, 2008.
- [22] Scott Doniger, Thomas Hofmann, and Joanne Yeh. Predicting CNS Permeability of Drug Molecules: Comparison of Neural Network and Support Vector Machine Algorithms. *Journal of Computational Biology*, 9(6):849–864, December 2002.
- [23] Alexander G. Dossetter, Edward J. Griffen, and Andrew G. Leach. Matched molecular pair analysis in drug discovery. *Drug Discovery Today*, 18(15-16):724–731, 2013. Publisher: Elsevier Ltd.
- [24] Jianxin Duan, Steven L. Dixon, Jeffrey F. Lowrie, and Woody Sherman. Analysis and comparison of 2D fingerprints: Insights into database screening performance using eight fingerprint methods. *Journal of Molecular Graphics and Modelling*, 29(2):157–170, September 2010.
- [25] Joseph L Durant, Burton A Leland, Douglas R Henry, and James G Nourse. Reoptimization of MDL Keys for Use in Drug Discovery. *Journal of Chemical Information and Computer Sciences*, 42(6):1273–1280, 2002.
- [26] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints, 2015. [_eprint: 1509.09292](#).
- [27] Matthias Fey and Jan Eric Lenssen. Fast Graph Representation Learning with PyTorch Geometric. *CoRR*, abs/1903.02428, 2019. [_eprint: 1903.02428](#).
- [28] Spencer M. Free and James W. Wilson. A Mathematical Contribution to Structure-Activity Studies. *Journal of Medicinal Chemistry*, 7(4):395–399, July 1964.
- [29] Toshio. Fujita, Junkichi. Iwasa, and Corwin. Hansch. A New Substituent Constant, π , Derived from Partition Coefficients. *Journal of the American Chemical Society*, 86(23):5175–5180, December 1964.
- [30] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun Mcglinchey, David Michalovich, Bissan Al-Lazikani, and John P Overington. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic Acids Research*, 40, 2012.

- [31] R.J. Gillespie. Fifty years of the VSEPR model. *Coordination Chemistry Reviews*, 252(12-14):1315–1327, July 2008.
- [32] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry, 2017. [_eprint: 1704.01212](#).
- [33] Rafaela Gladysz, Yves Adriaenssens, Hans De Winter, Jurgen Joossens, Anne-Marie Lambeir, Koen Augustyns, and Pieter Van der Veken. Discovery and SAR of Novel and Selective Inhibitors of Urokinase Plasminogen Activator (uPA) with an Imidazo[1,2-*a*]pyridine Scaffold. *Journal of Medicinal Chemistry*, 58(23):9238–9257, December 2015.
- [34] D Gonzalez-Arjona. Non-linear QSAR modeling by using multilayer perceptron feedforward neural networks trained by back-propagation. *Talanta*, 56(1):79–90, January 2002.
- [35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [36] Edward J Griffen, Alexander G Dossetter, and Andrew G Leach. Chemists: AI Is Here; Unite To Get the Benefits. *Journal of Medicinal Chemistry*, 63(16):8695–8704, 2020.
- [37] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, Josémiguel Josémiguel Hernándezhernández-Lobato, # Benjamín Sánchezsánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Alán Aspuru-Guzik. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Cent. Sci*, 4:268–276, 2018.
- [38] Johannes Hachmann, Roberto Olivares-Amaya, Sule Atahan-Evrenk, Carlos Amador-Bedolla, Roel S S Anchez-Carrera, Aryeh Gold-Parker, Leslie Vogt, Anna M Brockway, and Al An Aspuru-Guzik. The Harvard Clean Energy Project: Large-Scale Computational Screening and Design of Organic Photovoltaics on the World Community Grid. *J. Phys. Chem. Lett*, 15:12, 2011. Publisher: UTC.
- [39] Louis P. Hammett. The Effect of Structure upon the Reactions of Organic Compounds. Benzene Derivatives. *Journal of the American Chemical Society*, 59(1):96–103, January 1937.
- [40] Corwin Hansch, Toshio Fujita Vol, R B Roberts, D B Cowie, R Britten, P H Abelson, V Moses, O Holm-Hansen, J A Bassham, M Calvin, J Mol Biol, By Corwin Hansch, and Toshio Fujita. p - σ - π Analysis. A Method for the Correlation of Biological Activity and Chemical Structure. *J. Am. Chem. Soc.*, 86(8):1616–1626, 1964.
- [41] Moises Hassan, Robert D. Brown, Shikha Varma-O’Brien, and David Rogers. Cheminformatics analysis and learning in a data pipelining environment. *Molecular Diversity*, 10(3):283–299, August 2006.
- [42] Paul C D Hawkins, A Geoffrey Skillman, and Anthony Nicholls. Comparison of Shape-Matching and Docking as Virtual Screening Tools. *Journal of Medicinal Chemistry*, 50(1):74–82, 2006.
- [43] Jameed Hussain and Ceara Rea. Computationally Efficient Algorithm to Identify Matched Molecular Pairs (MMPs) in Large Data Sets. *Journal of Chemical Information and Modeling*, 50(3):339–348, March 2010.
- [44] John J. Irwin and Brian K. Shoichet. ZINC—a free database of commercially available compounds for virtual screening. *Journal of Chemical Information and Modeling*, 45(1):177–182, February 2005.
- [45] Shintaro Ishiwata, Wataru Kobayashi, Ichiro Terasaki, Kenichi Kato, and Masaki Takata. Structure-property relationship in the ordered-perovskite-related oxide Sr 3.12 Er 0.88 Co 4 O 10.5. *Physical Review B*, 75(22):220406, June 2007.
- [46] Paul Jaccard. THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1. *New Phytologist*, 11(2):37–50, February 1912.

- [47] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction Tree Variational Autoencoder for Molecular Graph Generation, 2019. [_eprint: 1802.04364](#).
- [48] Subha Kalyaanamoorthy and Yi-Ping Phoebe Chen. Structure-based drug design to augment hit discovery. *Drug Discovery Today*, 16(17-18):831–839, September 2011.
- [49] Steven Kearnes, Kevin Mccloskey, • Marc Berndl, Vijay Pande, and • Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *J Comput Aided Mol Des*, 30:595–608, 2016.
- [50] Ath Kehagias, G. Hollinger, and A. Gelastopoulos. Searching the Nodes of a Graph: Theory and Algorithms, 2009. [_eprint: 0905.3359](#).
- [51] Peter W Kenny and Jens Sadowski. Structure Modification in Chemical Databases. In Tudor I Oprea, editor, *Cheminformatics in Drug Discovery*, Methods and Principles in Medicinal Chemistry, pages 271–285. Wiley-VCH Verlag GmbH and Co. KGaA, 2005.
- [52] György M. Keserű and Gergely M. Makara. Hit discovery and hit-to-lead approaches. *Drug Discovery Today*, 11(15):741–748, August 2006.
- [53] Ross D. King, Jonathan D. Hirst, and Michael J. E. Sternberg. New approaches to QSAR: Neural networks and machine learning. *Perspectives in Drug Discovery and Design*, 1(2):279–290, December 1993.
- [54] Paul Labute, Chris Williams, Miklos Feher, Elizabeth Sourial, and Jonathan M Schmidt. Flexible Alignment of Small Molecules. *Journal of Medicinal Chemistry*, 2001.
- [55] Greg Landrum. RDKit: Open-source cheminformatics, 2021.
- [56] AG Leach, I Lukac, JM Zarnecka, AG Dossetter, and EJ Griffen. Matched Molecular Pair Analysis. In *Comprehensive Medicinal Chemistry III*, volume 3, pages 221–252. Elsevier, Amsterdam Oxford Cambridge, 3rd edition edition, 2017.
- [57] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. Publisher: MIT Press.
- [58] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated Graph Sequence Neural Networks, 2017. [_eprint: 1511.05493](#).
- [59] Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure-activity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015.
- [60] Andrea Mauri, Viviana Consonni, and Roberto Todeschini. Molecular Descriptors. In Jerzy Leszczynski, Anna Kaczmarek-Kedziera, Tomasz Puzyn, Manthos G. Papadopoulos, Heribert Reis, and Manoj K. Shukla, editors, *Handbook of Computational Chemistry*, pages 2065–2093. Springer International Publishing, Cham, 2017.
- [61] Kurt H. Meyer. Contributions to the theory of narcosis. *Transactions of the Faraday Society*, 33:1062, 1937.
- [62] H. L. Morgan. The Generation of a Unique Machine Description for Chemical Structures-A Technique Developed at Chemical Abstracts Service. *Journal of Chemical Documentation*, 5(2):107–113, May 1965.

- [63] Bruno J. Neves, Rodolpho C. Braga, Cleber C. Melo-Filho, José Teófilo Moreira-Filho, Eugene N. Muratov, and Carolina Horta Andrade. QSAR-based virtual screening: Advances and applications in drug discovery. *Frontiers in Pharmacology*, 9(NOV), November 2018. Publisher: Frontiers Media S.A.
- [64] Andrew Y. Ng and Michael I. Jordan. On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes, 2001.
- [65] Rick Ng. *Drugs: from discovery to approval*. Wiley Blackwell, Hoboken, New Jersey, third edition edition, 2015.
- [66] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems 32*, pages 8026–8037, 2019. Publisher: Curran Associates, Inc.
- [67] Jean-Louis Reymond. The Chemical Space Project. *Accounts of Chemical Research*, 48(3):722–730, March 2015. Publisher: American Chemical Society.
- [68] M. C. Richet. Note sur le Rapport Entre la Toxicite et les Propriretes Physiques des Corps. *Compt. Rend. Soc. Biol.(Paris)*, 45:775–776, 1893.
- [69] David Rogers and Mathew Hahn. Extended-Connectivity Fingerprints. *J. Chem. Inf. Model*, 50:742–754, 2010.
- [70] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.
- [71] David E. Rumelhart and James L. McClelland. Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, pages 318–362. MIT Press, 1987.
- [72] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning. *Physical Review Letters*, 108(5):058301, January 2012.
- [73] Samo Turk, Benjamin Merget, Friedrich Rippmann, and Simone Fulle. Coupling Matched Molecular Pairs with Machine Learning for Virtual Compound Optimization. *Journal of Chemical Information and Modeling*, 57(12):3079–3085, December 2017.
- [74] Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A. Hunter, Costas Bekas, and Alpha A. Lee. Molecular Transformer: A Model for Uncertainty-Calibrated Chemical Reaction Prediction. *ACS Central Science*, 5(9):1572–1583, September 2019.
- [75] Robert P Sheridan. Time-Split Cross-Validation as a Method for Estimating the Goodness of Prospective Prediction. *J. Chem. Inf. Model.*, page 8, 2013.
- [76] Robert P. Sheridan. Debunking the Idea that Ligand Efficiency Indices Are Superior to pIC50 as QSAR Activities. *Journal of Chemical Information and Modeling*, 56(11):2253–2262, November 2016.
- [77] Dagmar Stumpfe, Huabin Hu, and Jürgen Bajorath. Evolving Concept of Activity Cliffs. *ACS Omega*, 2019.

- [78] Govindan Subramanian, Bharath Ramsundar, Vijay Pande, and Rajiah Aldrin Denny. Computational Modeling of β -Secretase 1 (BACE-1) Inhibitors Using Ligand Based Approaches. *Journal of Chemical Information and Modelling*, 56(10):1936–1949, 2016.
- [79] Vladimir Svetnik, Andy Liaw, Christopher Tong, J. Christopher Culberson, Robert P. Sheridan, and Bradley P. Feuston. Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling. *Journal of Chemical Information and Computer Sciences*, 43(6):1947–1958, November 2003.
- [80] S. Joshua Swamidass and Pierre Baldi. Mathematical Correction for Fingerprint Similarity Measures to Improve Chemical Retrieval. *Journal of Chemical Information and Modeling*, 47(3):952–964, May 2007.
- [81] T.T. Tanimoto. *An Elementary Mathematical Theory of Classification and Prediction*. International Business Machines Corporation, 1958.
- [82] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition*. Elsevier Science & Technology Books, San Diego, CA, USA, 4th edition edition, 1998. OCLC: 892779912.
- [83] Thomas Unterthiner, Andreas Mayr, and J. Wegner. Deep Learning as an Opportunity in Virtual Screening. In *Workshop on Deep Learning and Representation Learning*, 2014.
- [84] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.
- [85] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer New York, New York, NY, 2000.
- [86] Daniel F. Veber, Stephen R. Johnson, Hung-Yuan Cheng, Brian R. Smith, Keith W. Ward, and Kenneth D. Kopple. Molecular Properties That Influence the Oral Bioavailability of Drug Candidates. *Journal of Medicinal Chemistry*, 45(12):2615–2623, June 2002.
- [87] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, 2018. [_eprint: 1710.10903](https://arxiv.org/abs/1710.10903).
- [88] Hugo O Villar and Mark R Hansen. Design of chemical libraries for screening. *Expert Opinion on Drug Discovery*, 4(12):1215–1220, December 2009.
- [89] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep Learning for Computer Vision: A Brief Review. *Computational Intelligence and Neuroscience*, 2018:7068349, February 2018. Publisher: Hindawi.
- [90] Gary W. Caldwell, Zhengyin Yan, Wensheng Lang, and John A. Masucci. The IC50 Concept Revisited. *Current Topics in Medicinal Chemistry*, 12(11):1282–1290, May 2012.
- [91] David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1):31–36, February 1988.
- [92] David J. Wood, Lars Carlsson, Martin Eklund, Ulf Norinder, and Jonna Stålring. QSAR with experimental and predictive distributions: an information theoretic approach for assessing model quality. *Journal of Computer-Aided Molecular Design*, 27(3):203–219, March 2013.
- [93] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: a benchmark for molecular machine learning †. *Chemical Science*, 1(2), 2018.

- [94] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2019. arXiv: 1901.00596v4.
- [95] Yuting Xu, Junshui Ma, Andy Liaw, Robert P. Sheridan, and Vladimir Svetnik. Demystifying Multitask Deep Neural Networks for Quantitative Structure-Activity Relationships. *Journal of Chemical Information and Modeling*, 57(10):2490–2504, 2017.
- [96] Geeta Yadav and Swastika Ganguly. Structure activity relationship (SAR) study of benzimidazole scaffold for different biological activities: A mini-review. *European Journal of Medicinal Chemistry*, 97:419–443, June 2015.
- [97] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, Andrew Palmer, Volker Settels, Tommi Jaakkola, Klavs Jensen, and Regina Barzilay. Analyzing Learned Molecular Representations for Property Prediction. *J. Chem. Inf. Model*, 59:23, 2019. Publisher: UTC.
- [98] Mihalis Yannakakis. The effect of a connectivity requirement on the complexity of maximum subgraph problems. *Journal of the ACM (JACM)*, 26(4):618–630, 1979. Publisher: ACM New York, NY, USA.
- [99] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation*, 31(7):1235–1270, July 2019.
- [100] Edward D. Zanders. Drug Discovery Pipeline Overview. In Edward D. Zanders, editor, *The Science and Business of Drug Discovery: Demystifying the Jargon*, pages 95–98. Springer International Publishing, Cham, 2020.
- [101] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, January 2020.