AI ACCELERATED COLLISIONAL CROSS SECTION PREDICTION FOR HIGH THROUGHPUT METABOLITE IDENTIFICATION

By

Kiyoto Aramis Tanemura

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Chemistry - Doctor Of Philosophy

2022

ABSTRACT

Metabolomics refers to the collective characterization of small organic molecules in a biological sample. While instrumentation and software continues to improve for metabolomics studies, the fraction of annotated signals in untargeted metabolomics experiments remain small. Translating features to metabolite identities present a major bottleneck, confounded by the lack of authentic standards to build comprehensive experimental databases. I illustrate the development of collisional cross section (CCS) prediction methods through deduction from theory and induction from available data. The theoretical CCS prediction involves multistep modeling of conformational ensemble followed by simulation of ion mobility. The advanced computational chemistry operations were automated using the AutoGraph conformational clustering protocol and implementation of the workflow in Snakemake. In a complementary approach, I applied a graph convolutional deep Bayesian neural net to predict CCS values and their uncertainty values. The quantified uncertainty was used to guide *ab initio* prediction of CCS values in an active learning strategy. The developed methodologies lay the foundation to a continuously refining *in silico* CCS library to aid in metabolite annotation.

I dedicate this dissertation to my supportive East Lansing community, colleagues, and friends whose excellence has inspired me to continue learning and improving.

ACKNOWLEDGEMENTS

I thank my advisor, Prof. Kenneth M. Merz Jr., for his mentorship in research. I thank Prof. Guowei Wei for his guidance over my research project at weekly meeting. I thank the colleagues in my research group who have contributed their time and expertise to my research and publications, including Dr. Susanta Das, Dr. Jun Pei, Dr. Madushanka Manathunga, Diego Sierra Costa, Akhil Shajan, Laleh Dinpazhouh, Mithony Keng, and Zhen Li. I thank current and former members of the research group who have also been integral to my scholarly development, including Dr. Ali Rahnamoun, Dr. Gaurav Sharma, Dr. Lin Song, Hongni Jin, Anthony Seitz, Dr. Chi Jin, Dr. Arka Sengupta, Dr. Quan Jiang, Dr. Zhuoqin Yu, Majid Jafari, and Andrew Candia. I also thank Prof. Heedeok Hong and Prof. Angela Wilson for evaluating my research as guidance committee members. I would like to thank the staff members for the Institute for Cyber-Enabled Research (ICER) for their outstanding efforts in maintain the High Performance Computing Cluster (HPCC) and addressing my queries and requests.

TABLE OF CONTENTS

Chapter 1: Introduction

1.1 Metabolomics

The *metabolome* refers to the comprehensive set of small organic molecules in a biological sample. It encompasses endogenous metabolites produced though cellular transformations, and exogenous metabolites consumed from the environment. The presence of endogenous metabolites may be inferred from knowledge of the proteins or genes of the sample. This is not the case for exogenous metabolites, or the *exposome*. Studying the metabolome provides description of the phenotype, and can yield insight in disease, therapy, and metabolic engineering. [1–7]

Metabolomics is among genomics and proteomics in the scientific paradigm of *omics*, which rely on holistic characterization of biological samples to draw insight. Owing to the complexity of the biological mixture and large volume of analytes to detect and quantify, omics studies rely on accurate, precise, and high-throughput analytical and informatic methodologies. Due to the relative standardization of subunit structures of the macromolecules investigated, genomics and proteomics have realized comprehensive, organismal-level characterization of genome and proteome. [8, 9]

Despite the interest in comprehensive characterization of the metabolome, limitations remain in metabolomics. Unlike other biomolecules with standard subunits, the number of possible metabolites is bound only by the number of thermodynamically stable structures, estimated to be $10^{33} - 10^{60}$ chemical structures depending on the assumptions applied. [10, 11] The immense chemical diversity also implies a broad collection of chemical/physical properties, such as polarity, volatility, and lability. Therefore, any one extraction method or instrumentation provides coverage over only a biased subset of the metabolome. [12, 13] The disparity between presently available technology and the goal of comprehensive metabolite annotation may be understood through the Human Metabolome Database (HMDB). [14, 15] The HMDB contains less than 5% of the estimated metabolite space across multiple

organisms. Meanwhile, only about 10% of the entries have authentic chemical standards commercially available. [16] While a large volume of metabolites remain for identification, majority of these unknown metabolites may evade confident identifications using authentic standards because the standards are unavailable.

A particular bottleneck presented in the untargeted metabolomics workflow is the annotation of features detected. The conventional untargeted metabolomics workflow is described as follows. Metabolites are extracted from biological samples. The complex mixture is separated by chromatography, then subjected to analytical methods such as mass spectrometry (MS) and nuclear magnetic resonance spectroscopy (NMR). [17] The collection of signals arising from a chemical structure is detected as a *feature*. The feature table is translated to metabolite identities by matching to databases of metabolites with known spectroscopic properties. The composition of the detected metabolites is then tied to biological insight. [18]

While informatic approaches may suffice to annotate metabolites already deposited in public databases (*known unknowns*), this is often a small fraction of the total signals observed in untargeted metabolomics experiments. The majority of features which remain unidentified represent the "metabolic dark matter". [17] Such elusive *unknown unknown* metabolites require a standard-free methodology for identification using *in silico* predicted values.

1.2 Spectral Prediction of Metabolites

Standard-free metabolite identification warrants the construction of a library of computationally predicted spectral properties for metabolites. The library must be larger than available experimental spectra, thereby computational methods require a high throughput. Meanwhile, accuracy and precision must approach that of experimental performance to distinguish many overlapping signals.

In silico prediction can be broadly divided into theoretical and empirical approaches. Theoretical methods employ molecular modeling and simulation to deduce spectral properties using first principles. Because the computed results are supported by physical principles,

the performance is not explicitly dependent on coverage of the metabolite space sampled. However, the many computational steps often require expertise and considerable computational resources. In contrast, empirical methods parameterize a mathematical model on available data to infer spectral values. While empirical methods are capable of rapid predictions due to delegating fine details of molecular modeling to statistical interpolation, their performance is dependent on the coverage, quality, and amount of spectral data available.

Among the challenges of studying the metabolome is throughput and relative sparsity of currently available data. While theoretical methods may return accurate predictions over sparsely sampled regions of the metabolic space, throughput is limited by its large computational demand. Meanwhile, empirical methods rapidly predict spectral properties, but are not guaranteed to generalize when extrapolating to sparsely sampled regions expected in the metabolite space. Development of *in silico* libraries for metabolite annotation may depend on informed integration of these disparate approaches to overcome their respective shortcomings.

1.3 Ion Mobility Spectrometry

Among the analytical techniques which offer high value target to develop predictive tools for metabolite identification is *ion mobility spectrometry* (IMS). IMS is an analytical separation technique in which ionized molecules are separated by mobility through an inert gas in a drift cell. The gas phase mobility of an ion depends on factors including its size, shape, charge, and polarity. [19] The mobility of an ion is its velocity through the drift cell normalized to the applied electric field, which is derived from the arrival time distribution. The mobility presents instrumental dependency, so instead the mobility is converted to collisional cross section (CCS), which can be understood as the rotationally averaged cross sectional area of an ion. The Mason-Schamp equation is used to convert mobility to CCS. [20] The CCS is a physicochemical value specific to each structure, thereby offer a valuable analytical descriptor for metabolites for improved annotation. [19] IMS is often coupled to MS in ion mobility mass spectrometry (IMMS) experiments, achieving an additional

degree of separation than by individual instruments while obtaining independent metrics for metabolite identification in a high throughput fashion. [21, 22]

Theoretical CCS values can be obtained at various degrees of molecular modeling. The *trajectory method* (TM) models long and short range interactions between the ion and drift gas molecule to numerically integrate the equations of motion. [23] The method is computationally intensive due to the many force evaluations performed over a continuous collision event and vast integration domain. Larger systems for which long range potentials are less influential may model only the short range potential using methods such as *elastic hard sphere scattering* (EHSS) method. [24] While the error in CCS introduced by omission of the long range potential has been reported to be less than 10% for select systems, the uncertainty introduced is not amenable for confident identification of metabolites from a mixture. The *projection approximation* (PA) method further simplifies CCS prediction by calculating the rotationally averaged geometric cross section. PA can yield CCS values with high degree of agreement for proteins. [25] However, a sizeable error can be introduced in cases for which surface concavity affects mobility.

The high accuracy achieved by the TM is therefore attractive for the context of metabolite annotation. TM is available in modern, open source implementations through software such as High Performance Collision Cross Section Calculation (HPCCS). [26] While agreement to experimental measurements is observed from TM predicted CCS values, this involves sizable computational effort for modeling the conformational ensemble expected during the trajectory down the drift cell. The arrival time is on the scale of milliseconds, thus flexible metabolites are expected to explore all thermodynamically accessible conformers; with each conformer contributing to the final CCS value. Thorough identification of local minima in the potential energy surface (PES) requires computation-intensive geometry optimization of many conformers. ISiCLE is a major *ab initio* CCS prediction software which encapsulates the entire CCS prediction workflow, including ensemble modeling of two dimensional chemical structures. [27]

To circumvent the complex workflow of ensemble modeling and CCS prediction by TM, machine learning algorithms have been extensively developed for CCS prediction. Simple regression models employ the support vector machine (SVM) algorithm. AllCCS achieves high degree of agreement to experimental measurements using SVM with only up to fifteen selected features. [28] The model associated with CCSBase is a SVM using the simple 42 features of the molecular quantum number (MQN) descriptor. [29, 30] Deep learning algorithms have been deployed to directly learn structure/property relationships for CCS prediction from an embedding of molecular structures using simplified molecular-input line-entry system (SMILES) representation. DeepCCS encodes the one-hot-encoded representation of the SMILES string using a convolutional neural net (CNN). [31] DarkChem utilizes CNNs in the encoder/decoder of a variational autoencoder, thereby enabling candidate structure prediction as well. [32]

Predicted CCS databases have been published for each CCS prediction models, however there are limitations. These models exhibit high accuracy over the metabolite space sampled, however the performance over yet to be annotated regions is unknown. Therefore, the uncertainty over the predicted CCS are heterogeneous. However the uncertainty of the predicted value is not quantified, limiting the confident assignment of CCS to metabolite by predicted values. Furthermore, SMILES is only one machine readable representation of molecular structure. Interpreting the grammar of SMILES is a nontrivial task for deep learning. More intuitive structural representations can be employed.

1.4 Machine Learning

Artificial intelligence (AI) is a field which aims to implement intelligent behavior in artificial systems by imitating natural intelligence. *Machine learning* (ML) is a branch of artificial intelligence (AI) which uses data and algorithms to parameterize mathematical models to accomplish tasks and behaviors not explicitly encoded. The delegation of finding solutions to various tasks to the parameterization is anthropomorphized as *learning* by the machine. ML dates back to 1959, when the term was incepted by Arthur Samuel of IBM. [33]

While ML as a field is not new, various technological advancements enabled the widespread applications of ML in what can be regarded as the Fourth Paradigm of Science. [34] The historic evolution of scientific paradigms: from empiricism, to analyses, to simulation; developed theoretical understanding of underlying mechanics to make predictions. The Fourth Paradigm of Data-intensive Scientific Discovery instead seeks to uncover underlying rules and theories from large volume of data. The reformulation of scientific discovery enabled by ML therefore is not constrained by the domain knowledge of the subject under investigation, but rather access to data and computational resources.

The disparate algorithms in ML are related to one another by the general tasks they accomplish. Simple, structured datasets are matrices of values, with each row corresponding to one *sample* (entry, instance, point, etc) and the columns are *features* (independent variables). *Supervised learning* algorithms operate over datasets with labels to predict, thereby producing models for quantitative prediction such as regression or classification. *Unsupervised learning* algorithms draw trends and generalizations from the distribution of unlabeled datasets. Other categories exist such as semi-supervised learning or active learning tasks. The performance of these algorithms depend on the features extracted from each sample, therefore traditional ML algorithms are not fully delineated from domain knowledge.

Artificial neural nets (ANN) free ML approaches from dependency to domain knowledge imposed by feature engineering. ANN are composed of interconnected *units* (artificial neurons, perceptron), which take the weighted sum of input features, pass the value to an activation function such as the logistic function, and output a value over the range of the activation function ($(0.0, 1.0)$ for the logistic function). The *single layer perceptron* (SLP) model organizes $N \in \mathbb{N}$ units in parallel to create a single *layer*. The weighted sum of the outputs of individual units is the prediction returned by the model. The Universal Approximation Theorem states any continuous function over an interval can be approximated by parameterizing a SLP model. [35] Therefore, given predictive features and enough data, a SLP can approximate any continuous function which maps the features to the label of

interest.

Identifying predictive features from raw data is a subject of domain knowledge. However, the procedure of embedding from raw data to a feature vector is a function, which can be approximated using the SLP model. In this manner, stacking layers in an ANN can automate feature extraction from raw data. Stacking several standardized layers in an ANN is known as a *multilayer perceptron* (MLP) model, which can compute increasingly abstract, hierarchical features from the raw data. This strategy to learn data representation through ANN of many layers is known as *deep learning* (DL). In chemistry, molecular structures determine all chemical and physical properties. Extracting predictive features by DL is therefore a general solution for many chemical problems.

Encoding chemical structures to machine readable form for DL determines the model architecture. Vector representation by molecular fingerprints provide a canonical representation in ML to utilize a MLP model. Fingerprint representations depend on a algorithm defined *a priori*, thereby are not guaranteed to capture substructures most predictive over the task and data. The characters of the SMILES representation of molecules can be embedded as inputs to CNN or recurrent neural nets (RNN). Learning the representation of SMILES may be complicated because several correct SMILES representations exist for one chemical structure, and the model must learn the grammar of SMILES embedding alongside extracting predictive features. The most practical means to communicate chemical structures among chemists is the graph representation using segment formula. Even images of segment formula has been used as inputs to CNN for property prediction. The *graph convolutional net* (GCN) has emerged as a prominent model for learning molecular graph representation. GCN are generalizations of CNN by performing convolution over sets non-standardized neighbors encoded in a *bond adjacency matrix*. The coherence of the graph representation between the chemical intuition of domain experts and as an exact input format for GCN renders the approach particularly attractive for learning molecular structures.

1.5 Overview

I highlight applications of ML algorithms to chemical problems to increase accuracy, automation, and throughput. A logistic regression classifier based scoring function is showed to detect identifiers of protein-protein interaction hotspots. I developed a conformational clustering algorithm, AutoGraph, to remove hyperparameter dependencies from modeling conformational ensembles. AutoGraph was integrated in an *ab inito* CCS prediction workflow, which accelerates CCS prediction via incremental refinement of structures using molecular mechanics, deep learning, and quantum mechanics. Finally, a graph convolutional deep Bayesian neural net was used to quantify uncertainty of predicted CCS values, thereby identify high value metabolites to annotate the CCS value by the *ab initio* workflow for the production of an *in silico* CCS library. I conclude with describing future work to implement a continuously refining CCS library for improving metabolite annotation.

Chapter 2: Refinement of pairwise potentials via logistic regression to score protein-protein interactions

Kiyoto Aramis Tanemura, Jun Pei, Kenneth M. Merz Jr.

2.1 Introduction

Protein-protein interactions (PPI) are present in the underlying mechanisms of virtually all biochemical processes. In terms of drug discovery, they present an alternative drug target to the traditional small binding pockets of enzyme active sites. [36, 37] For instance, the 90 kDa heat shock protein Hsp90 has been long pursued as a possible therapeutic target in cancer research. [38] While five molecular scaffolds have been investigated as competitive inhibitors of the N-terminal substrate-binding domain, the Hsp90 modulation by targeting the structurally relevant C-terminal domain (CTD) was underexplored. [39] Exploitation of the conformational dynamics natively induced by the binding of a client protein near the CTD led to the design of allosteric activators of Hsp90 ATPase activity. [40] Following this principle, the design of PPI inhibitors by mimicking the client protein interface has diversified the possible modes of Hsp90 inhibition. [41]

Experimental techniques for protein structural characterization such as X-ray crystallography, nuclear magnetic resonance spectroscopy, and cryo-electron microscopy have made available high-resolution data for proteins, including those of protein-protein complexes. [42] Protein-protein docking prediction as a computational method complements experimental techniques where experimental approaches do not suffice. Development of protein-protein docking methods advances the understanding of mechanisms of biologically important functions, as well as exploit their underlying PPI as a target for therapeutic agents.

Ab initio protein-protein docking is generally separated into two phases due to the complexity of the problem. These phases are the sampling of docking poses followed by their evaluation by means of a scoring function. [43] For the sampling phase, the protein subunits may be treated as rigid bodies, as is the case for the docking algorithms ZDOCK, FTDOCK,

and GRAMM. [44–46] Backbone flexibility can be modeled during the docking phase through normal mode analysis with modest computational cost, as exemplified by the flexible docking algorithms ATTRACT and SwarmDock. [47, 48] Soft surface or pseudoatomic representations are typically employed in both rigid-body or flexible docking protocols to smooth the potential energy surface and allow faster convergence to energy minima. Sidechain flexibility is commonly modeled in the refinement stage after sampling, as is the case in iATTRACT. [49]

Due to the copiousness of the predictions generated during the sampling phase, the scoring function must achieve high computational efficiency and must accurately assign low energy structures to low ranks in the scoring process. Scoring functions belong to broad categories of physics-based, knowledge-based, and machine learning (ML)-based. Physics-based scoring functions are widely used and include those used for ZRANK, ATTRACT, FASTCONTACT, FireDock, GalaxyTongDock, HawkRank, HADDOCK, and ClusPro. [50–57] Energy terms commonly include van der Waals, electrostatic, and desolvation potentials. Knowledge-based methods instead tend to apply Boltzmann inversion to the frequency of observed interatomic/interresidue distances to approxi- mate relative energies of PPI docking predictions. This class of scoring function include InterEvScore, SPIDER, and dDFIRE. [58–61]

PPI interfaces have been described as small patches of nonpolar and charged residues, with one residue pair near its center contributing to the majority of the energetic stabilization upon binding. [62] These energetic hot spots, as structural signatures, have been used to detect PPI interfaces. In the scoring function, Matrix of Low Coupling Energies (MLCE), nonbonding energies are calculated between each set of pairwise residues in the interacting protein pair and identifies likely PPI interfaces by Eigen decomposition. [63] Conservation and coevolution of residue pairs has been applied as a statistical approach to identify PPI hot spots in scoring functions including Evolutionary Trace (EVT) and InterEvScore. [58, 64] A recent comparison of MLCE and EVT revealed that the relative performances of each

scoring function depended on the biological function of the protein. [65] This highlights the observation that the development of a general PPI scoring function depends on devising an informed strategy to handle terms belonging to disparate scoring functions.

ML-based models provide just such a solution for refining or integrating terms belonging to a variety of scoring functions. Compared with classical scoring functions, ML-based methods have the advantage that they do not require prior assumptions between the structural data and protein-protein complex stability. This enables integrated processing of input data, as was exemplified by the PPI scoring function ProQDock and iScore. [66, 67] The rich set of supervised and unsupervised algorithms available in ML were applied to rationalize feature selection recursively to distinguish native-like ensemble of binding modes from decoys. [68] Further, the deep three-dimensional convolutional neural net, DOVE, was trained on PPI decoy set to automate the feature extraction process. [69] Meanwhile, ML-based models are frequently criticized for being a black-box alternative to well-defined scoring functions. This highlights the need to evaluate not only the performance of ML models but also the trends and insights it deduces from the data.

The random forest (RF) refinement methodology for native detection among decoys has been applied to protein-folding and protein-ligand decoy detections. [70, 71] In short, a dataset consisting of a native conformer and many decoy structures are featurized using conventional pairwise potentials such as Assisted Model Building with Energy Refinement (AMBER) force field and Knowledge-Based and Empirical Combined Scoring Algorithm (KECSA2) pairwise potentials. [72, 73] The extreme skew in representation of decoy and native structures of the dataset is mitigated by comparing the descriptors between the native and decoy structures. The balanced dataset is then suitable to train a RF model for binary classification. This methodology outperformed conventional programs for decoy detection in protein folding and protein-ligand. The novel methodology is further explored in the realm of scoring function for PPI prediction.

2.2 Methods

2.2.1 Decoy Set

The Critical Assessment of Scoring Function (CASF) - PPI decoy set was employed in this study, which consists of 273 systems with 2000 decoys each. [74] The decoys were generated previously by rigid-body docking using the FTDock software. [45] The CASF-PPI decoy set was more suitable than other decoy sets to train the ML-based scoring function because it removed artifacts and complications due to the docking protocol by using subunits of the bound PPI crystal structure as inputs to rigid-body docking. Therefore each PPI system consisted of one high quality native structure and decoy sets consistent in all respect but the binding mode.

2.2.2 Featurization of PPI Complexes

Let any PPI complex be described as an n-body system, and let all independent pairwise probabilities be known. The overall probability of the PPI complex is described as,

$$p_n = \prod_{i,j=1;i\neq j}^{n} c_{ij} \times p_{ij} \tag{2.1}$$

in which p_{ij} is the independent probability of particle pair i and j, and c_{ij} is its empirical scaling constant. As a PPI complex, the overall probability can be further be decomposed to bond, angle, torsion, and nonbonding interactions as follows,

$$p_{complex} = \left(\prod_{bond} c_{ij} \times p_{ij}\right)\left(\prod_{angle} c_{kl} \times p_{kl}\right)\left(\prod_{torsion} c_{mn} \times p_{mn}\right)\left(\prod_{nonbond} c_{pq} \times p_{pq}\right) \tag{2.2}$$

in which $c_{\alpha\beta}$ corresponds to the scaling constant and $p_{\alpha\beta}$ corresponds to the pairwise probability of bond (ij), angle (kl), torsion (mn), and nonbonding interaction (pq). The present work deals with decoy structures generated by rigid-body docking, thus bond, angle, and torsional probabilities are constant between a native structure and its decoys. The equation 2.2 is rewritten,

$$p_{complex} = C \times \left(\prod_{nonbond} c_{pq} \times p_{pq} \right) \tag{2.3}$$

for some constant C. Taking the natural logarithm yields the following,

$$\ln p_{complex} = \ln C + \sum_{nonbond} \ln \left(c_{pq} \times p_{pq} \right). \tag{2.4}$$

Pairwise nonbonding interaction probabilities were obtained using KECSA2 pairwise potentials. [72] The potential of nonbonding interactions between atoms A and B with distance r_i were described using the following Lennard-Jones type equation:

$$E_{AB}(r_i) = \epsilon_1 \left(\frac{\sigma}{r_i} \right)^{\alpha} - \epsilon_2 \left(\frac{\sigma}{r_i} \right)^{\beta} \tag{2.5}$$

in which parameters $\epsilon_1, \epsilon_2 \sigma, \alpha, \beta$ were obtained by the KECSA2 database. Then the potentials were translated to relative probabilities by Boltzmann distribution. Constant C and scaling factors c_{pq} are canceled upon generation of the comparison descriptors. Thus each structure was represented by a vector of 14028 features where each element represented a specific nonbonding interaction.

2.2.3 Generation of Comparison Descriptors

The skewed representation of native and decoy PPI complexes are balanced using a comparison method, previously applied to protein tertiary structure and protein-ligand decoy sets. [70, 71] The native structure was assumed to be more stable than the decoy, thus subtracting the logarithmic probability of the decoy from the native structure would result in a more positive vector and vice versa. By performing this subtraction, a balanced comparison dataset is generated, consisting of 4000 descriptors per system. A target label of '0' was appended for descriptors generated by decoy minus native, and '1' was appended for comparison descriptors of the other direction. The comparison descriptors were used as a balanced dataset to train the LR model.

2.2.4 Training and Evaluation of the LR Classifier

The LR classifier model (sklearn.linear_model.LogisticRegression) was trained on various fractions of of the shuffled dataset. [75] Default values were used for hyperparameters. Training and validation sets were standardized by zero mean and unit variance. Each model was subjected to five-fold cross validation to obtain training and validation accuracies. Several replicates of the operation were performed to ensure sufficient coverage of the decoy set partitioned to the training set. Test accuracies were computed from the models refit on the training/validation sets. Accuracy is defined as,

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.6}$$

for the count of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions. Accuracy was amenable for evaluating the performance of the LR classifier because the representation of target label was balanced.

2.2.5 Scoring and Ranking of Decoy Sets via LR Classifier

The PPI complexes for each system were ranked as follows. Let $\mathbf{x}_i$ be a descriptor of structure i. Then comparison descriptors were generated by $\mathbf{x}_i - \mathbf{x}_j$ for all $j \neq i$, thereby generating 2000 comparison descriptors. These comparison descriptors were classified by the LR classifier as either '0' or '1'. Finally, these labels were summed to provide a score for x_i, in which the greater value was predicted to be the more native-like structure. Once all structures were scored, then the structures were ranked by their score in descending order.

2.2.6 Evaluation of Ranks Assigned by Scoring Function

Metrics used to evaluate the scoring functions include success rate (SR), modified success rate (Y), native ranking, first root mean square deviation (RMSD), first decoy RMSD, and Spearman correlation coefficient of ligand RMSD and rank. CAPRI criteria were used to define near-native structures. [76]

Performances of the LR scoring function were compared to the scoring functions: AT-

TRACT, dDFIRE, FASTCONTACT, and ZRANK. [50, 52, 60, 77] The scores for each scoring function were previously calculated and included with the CASF-PPI decoy set. [74]

Success Rate

SR is the probability of finding a near native structure in the top N predictions. Let $X = \{X_1, X_2, \cdots, X_n\}$ be the set of all PPI systems in the dataset. Let $h(X_i, N)$ be the number of near native structures in the top N predictions of system X_i. Then we write the success rate as,

$$SR(N) = \frac{\sum_{i=1}^{n}(H(X_i, N))}{n}; H(X_i, N) = \begin{cases} 1 & \text{if } h(X_i, N) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

Modified Success Rate

Y takes into account the fraction of near-native structures identified in the top N predictions. [78] Additionally, it assigns a higher score to the lower ranking of near-native structure. It is defined as,

$$Y = \frac{\sum_{i=1}^{n} F(X_i)}{n};\ F(X_i) = \frac{\sum_{j=1}^{N}(1 + top_j)}{h(X_i, |X_i|)};\ top_j = rank_j^{-1} \tag{2.8}$$

Spearman Correlation Coefficient of ligand RMSD and rank

The ideal funnel shape between ligand RMSD and rank of scoring function are quantified using Spearman correlation coefficient (ρ). A more positive ρ is ideal, as it suggests the lower rank is associated with lower ligand RMSD and vice versa.

Fold Enrichment of Near-Native Predictions in Top Ranked Structures

We measure the representation of various qualities of predictions in the top ranked structures using the mean of the quotient of observed near-native prediction divided by its expectation E, and refer to it as fold enrichment FE. The expectation is the probability of arbitrarily choosing a near-native structure by sampling exactly one structure from all predictions of a given PPI system. We define the fold enrichment as,

$$FE(X) = \frac{\sum_{i=1}^{n} \frac{h(X_i,1)}{E(X_i)}}{n};\ E(X_i) = \frac{h(X_i,|X_i|)}{|X_i|} \tag{2.9}$$

2.2.7 Analysis and Selection of Feature Coefficients

Median values of coefficients assigned to each feature were obtained from LR classifiers trained on 0.99 of the data. The features consist of pairwise nonbonding interactions of heavy atoms of residues. The residues were assigned to the following subjective categories: anionic, cationic, polar, nonpolar, aromatic, flexible, and small (Table A1). The interactions of the residues were assigned by their categories (e.g. cationic-anionic). Distributions of the types of interactions were assessed.

Features with median coefficients of the greater magnitude were considered more salient. Subsets of features were taken as fractions of top salient features, and were used to train LR classifiers. The performance of the LR classifiers with reduced dimensions were assessed via aforementioned metrics.

2.2.8 Flexible Docking of Weng Benchmark 5.0 by ATTRACT

An independent decoy set was generated as a benchmark to assess the performance of various scoring functions. Unbound subunits of the Weng Benchmark 5.0 (BM5) were subjected to ATTRACT flexible docking with the iATTRACT interface refinement. [47, 49, 79] Bash script for performing docking was obtained from the ATTRACT web interface. [80] Because the weights of terms in the ZRANK scoring function were fitted on structures in the Weng Benchmark 1.0, structures from Benchmark 1.0 were removed for fair comparisons between scoring functions. [50, 81] Likewise, systems in BM5 were omitted if either one of the subunits exhibited a sequence identity of greater than 30% to any protein subunit in the CASF-PPI data set, resulting in a maximum sequence identity of 30% between protein subunits in BM5 and CASF-PPI. One thousand structures were generated using five normal modes. The complexes with the following PDB IDs were not further considered because they required repair of missing atoms in the input files to perform docking: 1F51, 1F6M, 1RLB,

1SYX, 2CFH, 4FZA, and 4GAM. Out of 69 systems considered, 40 structures contained at least one acceptable structure.

2.3 Results and discussion

2.3.1 LR Classifier Achieved High Accuracy on CASF-PPI Test Set

A learning curve was plotted for the LR classifier trained on various fractions of the decoy set (Figure 2.1). As the fraction of data used as the training set was increased to 0.99, the validation and test score approached accuracy of 0.99. The small differences between training accuracy and validation or test accuracy illustrate the proficient model performance generalized to the remainder of the CASF-PPI decoy set. The narrow range in each accuracy indicated the model performance was stable. Near optimal performance of the LR classifier on the decoy set was observed with a training set fraction of 0.7 or greater.

2.3.2 Analysis and Selection of Salient Features

The input data was standardized to zero mean and unit variance. Thus the magnitude and sign of coefficients provide a measure for salient features in detecting native structures. The coefficients plotted in decreasing order displayed a logit shape, with the magnitudes rapidly decreasing toward the center of the distribution (Figure 2.2). This suggests there was a relatively small subset of highly salient features, while the majority of features contributed moderately to the classification task. The range of coefficients for each feature was narrow, suggesting the similarity in coefficients between models.

To generalize the types of interactions contributing to the classification, the interacting residues were categorized to broad classes and the type of interacting residues were recorded. The density plot displayed the representation of a given type of interaction over the features ordered by their coefficient values (Figure 2.2). Qualitatively, interactions between charged residues displayed the highest representations at the ends of the distribution, representing coefficients with greater magnitude and corresponding with features with greater effect on the classification. This is consistent with ionic interactions as the strong, specific, and dynamic nonbonding interaction characteristic to PPI. [82] As expected, opposing charges

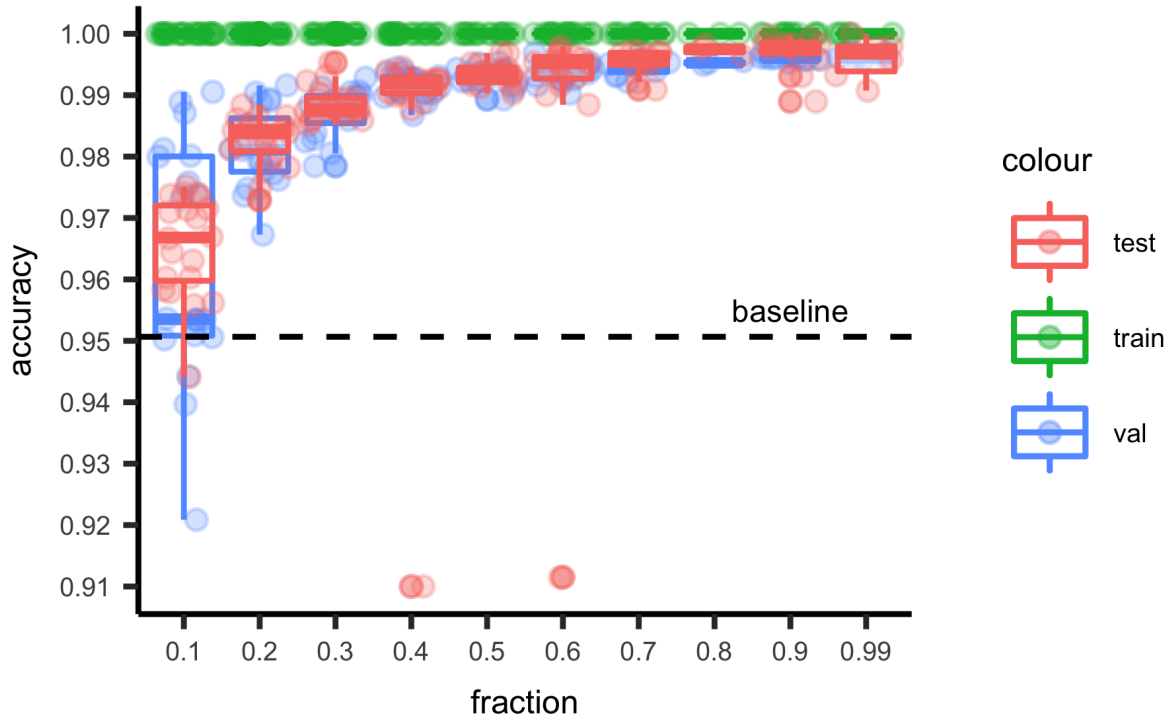

Figure 2.1: The distributions of training (green), validation (blue), and test (red) accuracies of the LR classifiers trained on various fractions of the decoy set. The baseline is the accuracy achieved by linear, unweighted sum of KECSA2 potentials with no LR refinement. Training and validation accuracies were obtained by taking the mean of the five-fold cross validation results. Test accuracy was calculated from LR classifiers refit on the training and validation set.

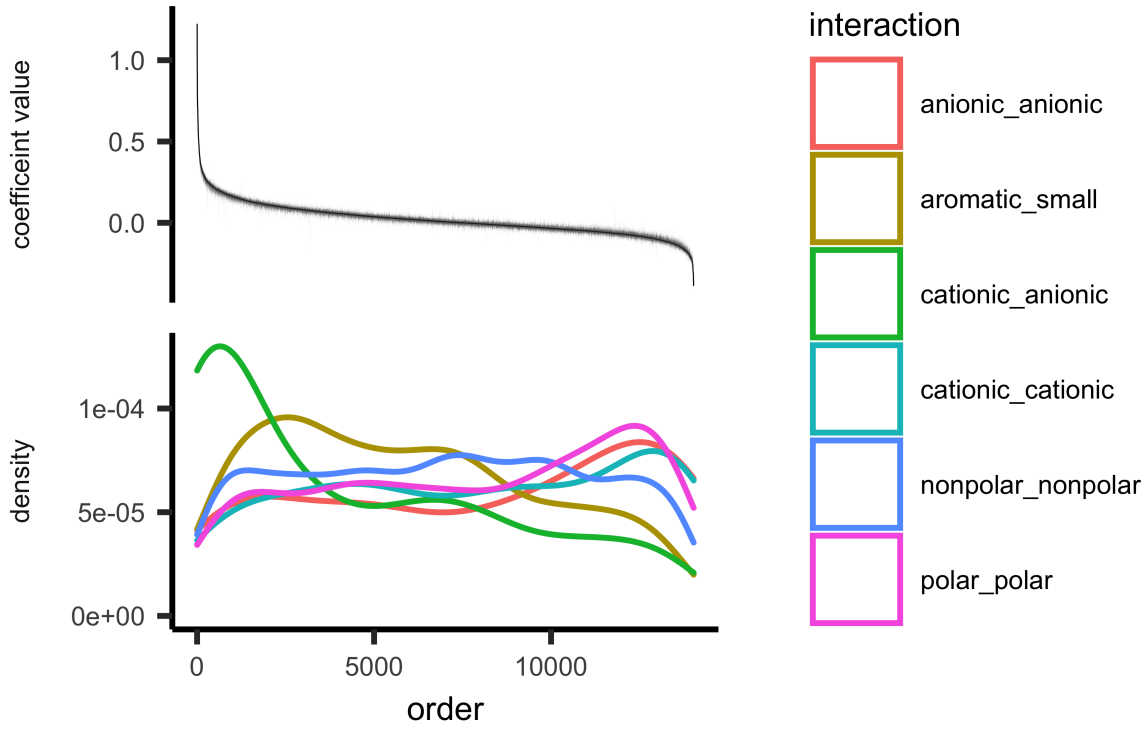

Figure 2.2: (top) Median coefficients for each feature in descending order. The curve displayed a logit shape, with the magnitude of the coefficients rapidly decreasing toward the center of the distribution. Maximum and minimum for each coefficient were plotted as a gray ribbon. (bottom) Density of various types of interactions applied to the coefficients ordered in decreasing order.

were favored by having a greater density on the side of positive coefficients, while like charges had high representation for negative coefficients.

Aromatic-small residue interactions were also notable on the positive end of coefficients. The shape complementarity achieved by hydrophobic residues of opposing sizes has been attributed a key factor for affecting PPI. [83] The relatively high representation may signify the importance of shape complementarity of hydrophobic residues between interacting proteins. The distribution is contrasted from the fairly even distribution of nonpolar-nonpolar interaction, further emphasizing the importance of shape complementarity.

Furthermore, the present LR classifiers represented polar-polar residue interaction with negative coefficients. A greater change in potential energy upon the exclusion of water molecules from PPI interface upon binding would be expected if favorable interaction between solvent and polar residue were absent. The higher density of polar-polar residue with negative coefficients suggests polar residues are underrepresented in native PPI interfaces. Thus, the coefficients may implicitly suggest that the desolvation energy to be a contributor to the detection of native PPI complexes.

In summary, the LR classifier appeared to prioritize charge and geometric complementarity while disfavoring polar-polar interaction. A more complete plot is available in Appendix (Figure A1)

Dimensionality reduction was pursued by training logistic regression classifiers with features associated with coefficients of greater magnitude. Specifically, the features were ordered by the magnitude of the median coefficient, then various fraction of the salient features were used as input data. The performance of models trained on 0.9 of the data set are reported (Table 2.1). Test accuracy and native ranks appeared to peak when around 0.1 of features were selected. Meanwhile, first decoy RMSD was constant between the fractions. The improvement in performance may be due to reducing noise arising from superfluous features. Refer to Figures A2-A4 for the full performance metrics.

The performance of the LR scoring function trained on the 0.1 top features (consisting

Table 2.1: Performance metrics of the LR classifier trained on various fractions of top salient features. For each quantity, the mean, 25th percentile (lower) and 75th percentile (upper) are reported. Four digits are reported all values except for lower/upper quantiles of native rank, which are natural numbers.

fraction	test accuracy			native rank			first decoy RMSD (Å)		
	mean	lower	upper	mean	lower	upper	mean	lower	upper
1.00	0.9969	0.9968	0.9984	7.216	1	3	22.33	12.63	30.88
0.50	0.9993	0.9990	0.9997	2.344	1	1	23.68	13.03	31.92
0.40	0.9990	0.9992	0.9999	3.020	1	1	23.96	13.11	32.46
0.30	0.9994	0.9995	0.9999	2.285	1	1	23.96	13.80	31.92
0.20	0.9995	0.9997	0.9999	2.044	1	1	22.63	13.10	31.08
0.10	0.9997	0.9996	0.9998	1.665	1	1	22.34	12.91	31.10
0.05	0.9992	0.9990	0.9997	2.585	1	2	23.03	13.23	32.99

of 1403 features in total) was selected for further investigation. By use of a simple ML model coupled with the use of only 0.1 of the most salient features, we arrive to a scoring function, which ranks thousands of structures on the order of minutes. These features are summarized in Figure A5. The performance of the new scoring function was benchmarked.

2.3.3 The LR Scoring Function Was Sensitive towards Native Structures While Less Responsive to Near-Native Structures

The performance of the LR scoring function trained 0.1 top features was compared to those of conventional scoring functions (Figure A2). The SR of the LR scoring function was relatively high for the most stringent threshold quality, which considered only the native structure as a near-native structure. There were little improvements in SR as the threshold quality was relaxed down to acceptable predictions. Unlike other scoring functions, the LR scoring function displayed an early saturation of SR at about $N = 10$. A plateau in Y accompanies this trend, in which the LR scoring function displayed little improvement above $N = 10$ for thresholds *native* and *high*.

The sensitivity of the LR classifier to native structures was further exemplified in the distribution of native ranks between the various scoring functions (Table 2.2). The LR scoring function yielded the lowest mean native ranks compared to the other scoring functions. The RMSD of low ranking structures illustrated a different trend. The mean of the first RMSD

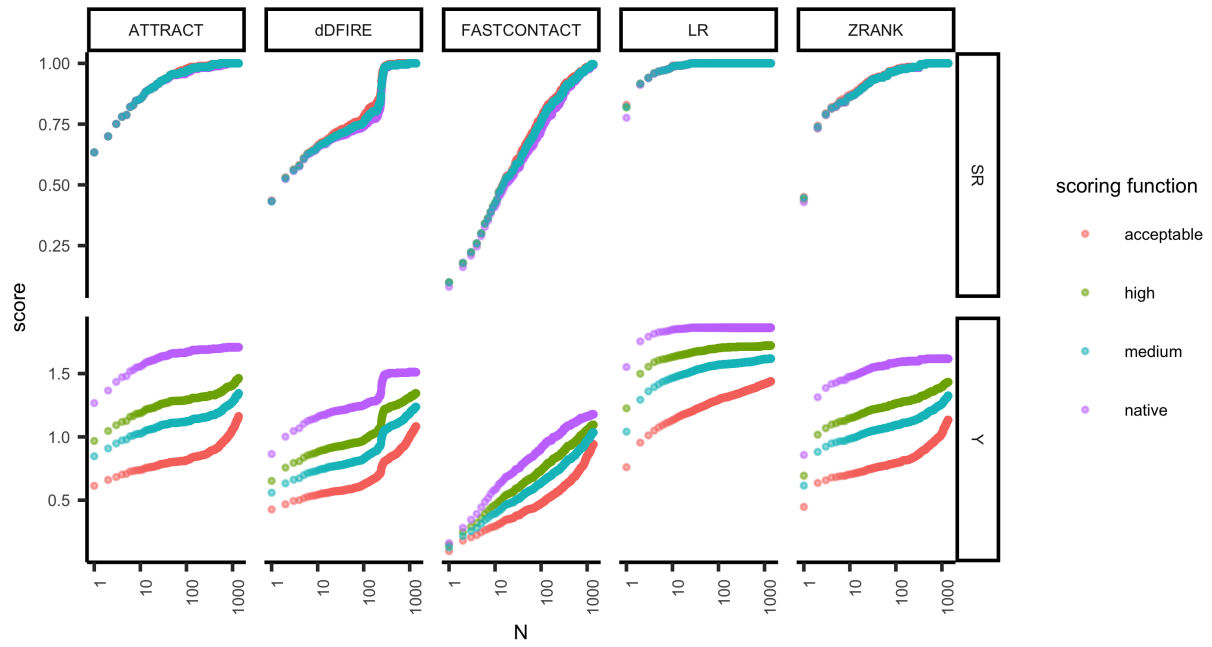

Figure 2.3: Comparison of scoring functions by success rate (top) and modified success rate (bottom) for various threshold of near-native structures.

for LR scoring function was low due to the superior native ranking compared to other scoring functions. Yet the insensitivity of the LR scoring function to near native decoy structures was apparent in the distribution of the first decoy RMSD, in which the LR scoring function displayed the greatest mean compared to other scoring functions. ZRANK notably exhibited the greatest performance for assigning near native structures to low ranks. Plots of the distributions are available in SI (Figure A6).

A similar comparison was present in the Spearman correlation coefficient between ligand RMSD and rank (Figure 2.4). If only structures with ligand RMSD up to 5 Å were considered, all scoring functions displayed ρ near 0.5. While other scoring functions still preserved a distribution centered at a positive value when structures up to 10 Å were considered, ρ for ATTRACT and the LR scoring functions returned to a distribution centered at 0.0. All distributions were centered at 0.0 if structures up to 20 Å were considered. The relatively early erosion of correlation for ATTRACT and LR scoring functions emphasize they are less responsive to near-native decoy structures than other scoring functions. In the context

Table 2.2: Performance metrics of the LR classifier trained on 0.1 top salient features in comparison to other scoring functions (SF). For each quantity, the mean, 25th percentile (lower) and 75th percentile (upper) are reported. Four digits are reported all values except for lower/upper quantiles of native rank, which are natural numbers.

SF	native rank			first RMSD (Å)			first decoy RMSD (Å)		
	mean	lower	upper	mean	lower	upper	mean	lower	upper
LR	1.665	1	1	5.343	0.000	0.000	22.34	12.91	31.10
ATTRACT	16.70	1	3	7.256	0.000	12.98	19.90	9.952	28.89
dDFIRE	68.06	1	109	12.11	0.000	24.07	15.58	1.496	27.85
FASTCONTACT	129.2	5	116	15.34	2.064	26.43	15.90	2.744	26.58
ZRANK	13.52	1	3	6.047	0.000	3.336	9.667	0.982	15.75

of docking protocols which may generate near native predictions but not necessarily close matches to the native structure, the sensitivity to near native structures exhibited particularly by ZRANK may be more desirable as the accompanying scoring function for rigid-body docking. The superior performance of the LR scoring function on detecting native structure suggested its utility will lie in flexible docking predictions, in which the structures are predisposed to be of higher quality than its early stage rigid body counterparts. This led to the independent assessment of the LR scoring function on decoys generated by ATTRACT flexible docking protocol with iATTRACT interface optimization. [47, 49]

2.3.4 The LR Scoring Function Performed Competitively on ATTRACT PPI Docking Predictions

We confirmed the performance of the LR scoring function generalizes on the CASF-PPI dataset due to its performance on the validation and test sets partitioned from the CASF-PPI dataset. The performance was further assessed on ATTRACT flexible docking predictions of PPI complexes in Weng Benchmark 5.0. [79] The independent dataset serves as assessment for whether the performance of the LR model was inflated by correlation in docking protocol between train and test sets, or biases arising by possible artifacts in docking which were not present in the native structure.

The SR and modified SR were determined for predictions ranked by the LR scoring

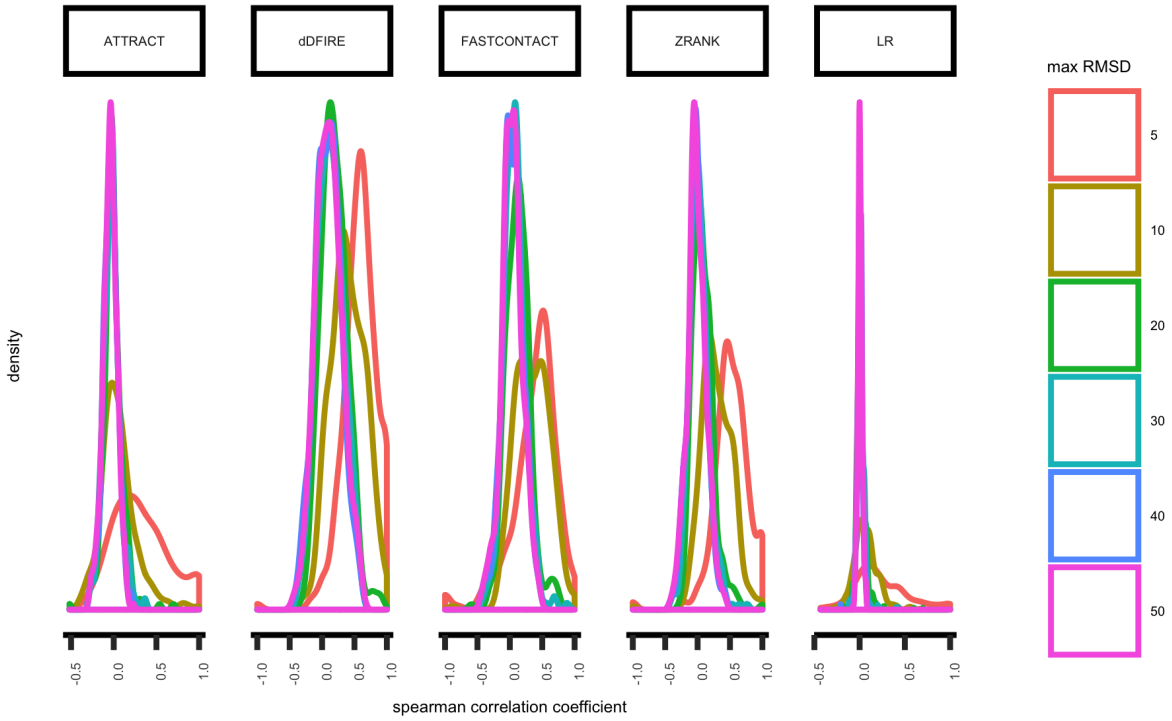

Figure 2.4: Comparison of scoring functions via Spearman correlation coefficient between ligand RMSD and rank assigned by scoring function under various maximum RMSD values.

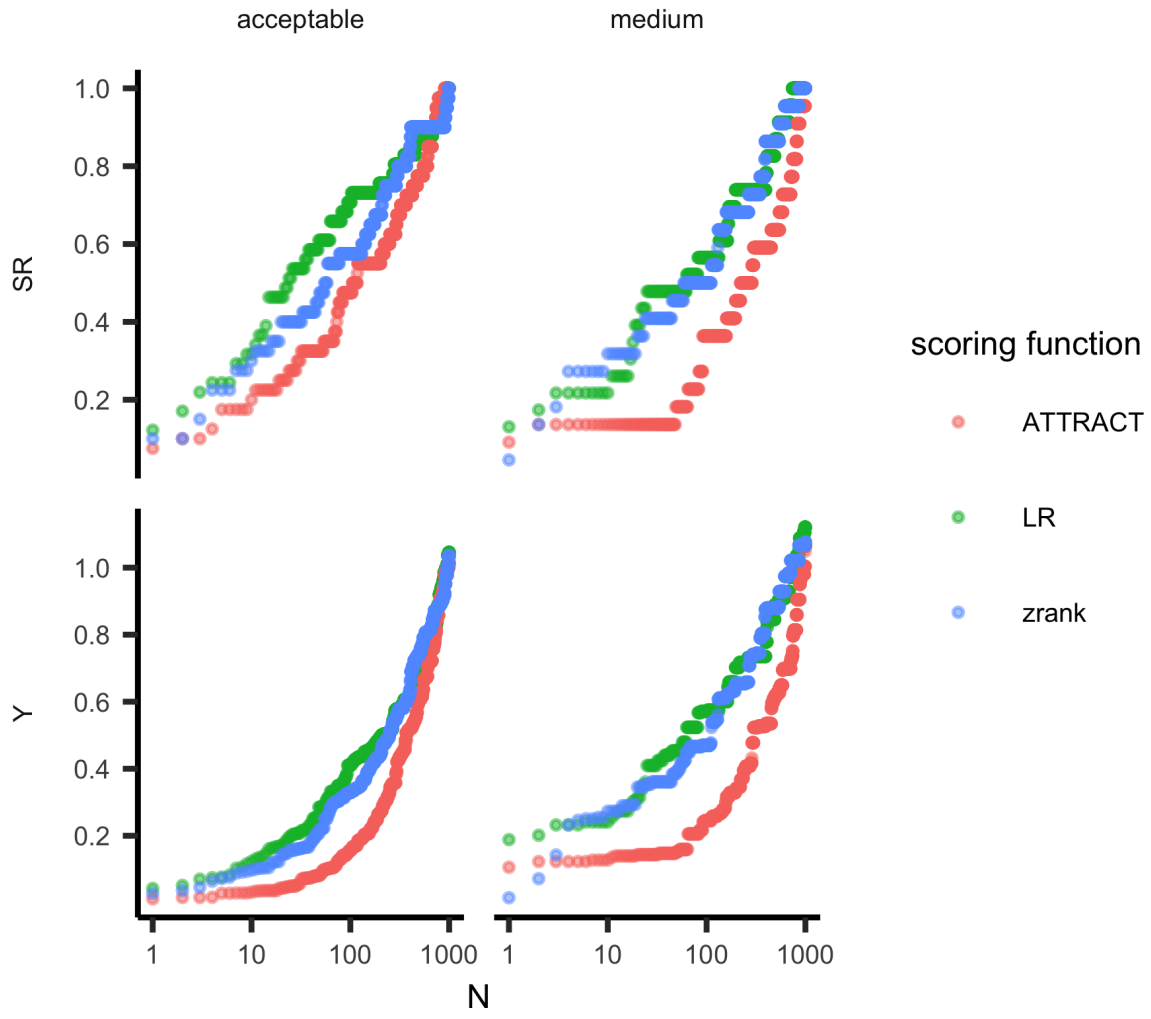

Figure 2.5: Comparison of scoring functions via success rate (top) and modified success rate (bottom) using various thresholds for near-native structures.

function, ATTRACT, and ZRANK (Figure 2.5). ATTRACT and ZRANK were selected for comparison because they per- formed competitively on the CASF-PPI data set. The LR scoring function performed comparably to ATTRACT and ZRANK, with a moderate lead in the top 100 candidates. LR scoring function appears competitive to other scoring functions on a set of realistic docking predictions.

While metrics of the top predictions (low N) are crucial for the performance of scoring functions, the differences are obscured due to the maximum value occurring at saturation ($N = 1000$). To assess the performance of the top predictions, we calculated the fold

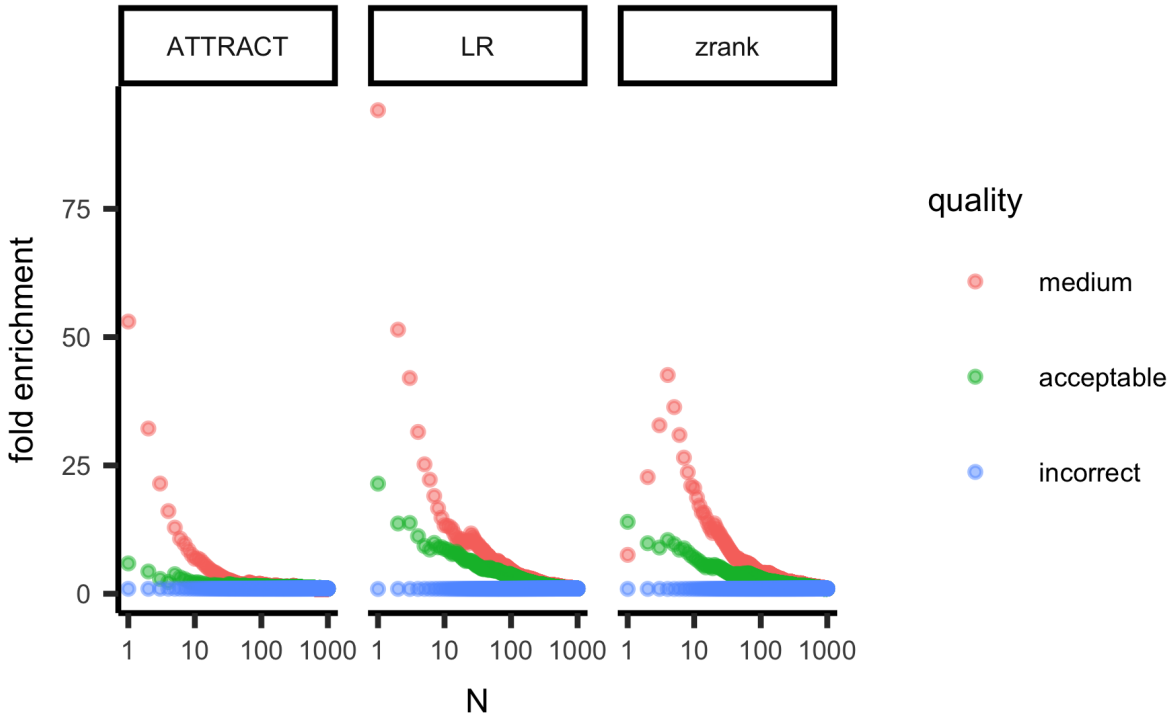

Figure 2.6: Fold enrichment in the top scoring structures is reported for various scoring functions (SF) and quality of prediction. Sample sizes by quality: $n_{medium} = 22$, $n_{acceptable} = 40$, $n_{incorrect} = 69$.

enrichment of the number of near-native structures detected in the top N predictions. The fold enrichment at a given quality threshold is the mean quotient of the fraction of observed near-native structures and the probabilities of arbitrarily choosing at least one near-native structure.

The fold enrichment is summarized for each scoring function in Figure 2.6. In the case of each scoring function, the elevation of medium and acceptable near-native predictions over incorrect structures illustrate the scoring function's sensitivity toward near-native structures. While the representation of near-native structures by LR and ZRANK are generally matched, the LR scoring function shows a greater representation of near-native structures as the top prediction. Combined with the SR, this illustrates the LR scoring function has a sensitivity to near-native structure which is competitive with ZRANK, coupled with a greater propensity to find the near-native structure as the top prediction.

2.4 Conclusions

The LR scoring function of PPI prediction illustrates the RF refinement of pairwise potentials extend to protein quaternary structure prediction and performs competitively to conventional scoring functions for the task of native detection among PPI decoys. The salient features were consistent with terms present in the various physics-based scoring functions. The utility of the scoring function was highlighted on ATTRACT flexible docking predictions, in which the representation of high quality structures was greater at the top ranked predictions compared to other scoring functions.

Chapter 3: AutoGraph: Autonomous Graph-Based Clustering of Small-Molecule Conformations

Kiyoto Aramis Tanemura, Susanta Das, Kenneth M. Merz Jr.

3.1 Introduction

Accurately modeling the distribution of equilibrium conformers is prerequisite in predicting the microscopic and macroscopic properties of flexible molecules. Obtaining a conformational ensemble is foundational to calculating average properties of molecular systems. [84] Methodologies such as ensemble docking of protein-ligand systems[85], three dimensional quantitative structure-activity relationship[86], and constructing Markov state models (MSM) from molecular dynamics trajectories[87] rely on sufficiently sampling from the conformational ensemble.

Many methods, algorithms, and their variants exist for conformation generation. [88] The objective of conformation generation protocols is to identify many equilibrium conformers at local minima of the potential energy surface (PES), which would be major contributors among all thermally accessible conformations. This may involve sufficient sampling of nonredundant conformations, followed by refinement of those conformations to local energy minima. Sequential methods such as molecular dynamics and Monte Carlo simulated annealing combine sampling and scoring of conformers to return physically informed, low energy conformers, however are generally computationally intensive compared to knowledge based methods. [89, 90] Knowledge based methods such as OMEGA and ETKDG algorithms narrow the search space by using the distributions of observed dihedral angles and ring structures from crystallographic databases. [91, 92] The rapid conformation generation by such algorithms should be followed up with physically informed structure refinement.

Geometry optimization by *ab initio* calculations converge to low energy conformations, however the high computational cost limits its applicability to the numerous conformations which need to be sampled. Recent development and benchmarking of machine learning based

potentials such as ANI-2x have prompted its utilization for certain high throughput quantum chemical applications. [93, 94] For example, ANI-1ccx potentials were used to accelerate the refinement of generated conformers in a quantum mechanical (QM) NMR spectral prediction workflow. [95, 96] The conformers optimized by these models, however, generally do not converge to the same local minima as *ab initio* methods. Hence, conformational clustering becomes important in narrowing the number of "full" QM geometry optimization calculations required to obtain a representative set of conformers to estimate the conformational ensemble.

Clustering is a task in unsupervised machine learning, in which individual data are coarse grained into disjoint groups. Clustering is used for purposes such as auto-label generation, dimensionality reduction, image segmentation, and visualization of data. To date, numerous clustering algorithms and their variants have been developed and deployed across disciplinary lines. [97, 98] Many conformational clustering algorithms have also been evaluated. [99–103] A majority of these algorithms require the number of clusters or threshold values defined *a priori*, though these hyperparameters vary by the data under evaluation and its choice may be nontrivial. [104] Unless automated, the iterative guess and check of hyperparameters can render the clustering protocol into one requiring supervision, thus limiting its throughput and integrity from user bias.

Highly automated conformational clustering protocols which do not require the number of clusters or threshold value be predefined would be advantageous for applications such as high throughput metabolomics. NMRCLUST is one such algorithm, readily available in its implementation through UCSF Chimera. [105, 106] NMRCLUST (NC) subjects conformations to the average linkage algorithm for hierarchical clustering. Clusters are merged to minimize a penalty function in order to balance between controlling the average spread within clusters and populating the clusters with as many conformers as possible. The algorithm was assessed for clustering small organic molecules. [100] The Dynamic Tree Cut (DTC) algorithm is an alternative strategy to define clusters on dendrograms using a dynamic threshold value. [107] The DTC algorithm was applied to identify conformational

clusters among drug-like molecules. [102] Unlike the aforementioned hierarchical clustering methods, the Representative Conformer K-means (RCK) algorithm employs the popular K-means algorithm to cluster conformers. [102] The protocol automates the determination of the number of clusters K by iterative calculations to find the value for K which maximizes the mean squared distance between clusters.

Here we present an autonomous graph based conformational clustering algorithm named AutoGraph. AutoGraph processes the atomic root-mean-squared deviation (RMSD) matrix between conformers into an affinity matrix using a generic Gaussian kernel. The matrix is processed as a graph object and its nodes are clustered using the Louvain algorithm, which does not require number of clusters or thresholds be predefined. [108] We estimate the conformational ensembles for *O*-succinyl-L-homoserine and nicotinamide adenine dinucleotide as simple examples before exploring the conformational graphs of 200 representative metabolites.

3.2 Methods

3.2.1 Definitions

Let there be $n \in \mathbb{N}$ conformers of exactly one molecule.

- A graph $G = (V, E)$ consists of a set of vertices V and a set of edges E.
- A **distance matrix** is an $n \times n$ matrix recording the pairwise dissimilarity between all conformers.
- An **affinity matrix** is an $n \times n$ matrix storing the pairwise similarity between conformers.
- A binary **adjacency matrix** is an $n \times n$ matrix with 1 indicating the presence of an edge and 0 otherwise.
- We define a **filtered matrix** of matrix M be the element-wise product between M and an adjacency matrix.

3.2.2 The AutoGraph Conformational Clustering Algorithm

The AutoGraph protocol is described (Figure A7). Atomic root mean squared deviation (RMSD) are computed comprehensively between n structures and stored in a symmetric $n \times n$ distance matrix. The Kabsch algorithm is used for finding the minimum pairwise RMSD. [109] An affinity matrix is calculated by applying a generic radial basis function, $\varphi(r) = \exp(-r^2)$ for distance r. A threshold value is applied to remove edges with low valued weights from the affinity matrix. This is performed by a breadth first search on the affinity matrix to find the maximum threshold value that does not produce disjoint components (i.e. given any pair of conformers, there exists a path connecting the two nodes in a graph.) An adjacency matrix is produced by applying this threshold to the affinity matrix. Let us refer to this threshold as *the adaptive threshold* or *threshold* in this manuscript. The resulting filtered affinity matrix encodes an undirected, weighted graph $G = (V, E)$ consisting of vertices V and edges E. Clusters are detected by applying the Louvain algorithm to the filtered affinity matrix. [108] The lowest energy conformer is reported for each cluster as its representative conformer.

3.2.3 Benchmark Conformational Clustering Algorithms

Among the goals of developing AutoGraph is to automate conformational clustering without introducing system specificity. For this reason, we select highly automated conformational clustering algorithms which do not assume specific threshold values or number of clusters to compare against AutoGraph. NMRCLUST, Representative Conformer K-means, and the Ward hierarchical clustering/Dynamic Tree Cut algorithm were chosen by this criterion. We implemented each algorithm in a sequential, CPU manner using only common Python packages (Numpy, Pandas, SciPy). [110–112] Only general descriptions or modifications are provided. Their original publications should be referenced for full descriptions of each algorithm.

NMRCLUST

The previously described NMRCLUST algorithm was implemented by us. [105] The

distance matrix is obtained in the same manner as AutoGraph. The average linkage was calculated between all conformers at each step. Clusters having the minimum average linkage were merged. Once no singleton clusters were present, the average spread of clusters were calculated. Each iteration of clustering was evaluated using a penalty function, with the normalized average spread and number of clusters as inputs. The final clustering was determined by the iteration having the minimum penalty value.

Unfortunately, the average spread as described in the original publication cannot be calculated if a singleton cluster is present. Few systems examined in this study contained a singleton cluster until the last merge into one cluster containing all conformers, which was returned as the clustering result. We encourage use of the NMRCLUST algorithm implemented in UCSF Chimera as an alternative. [106]

Representative Conformer K-means

The previously described RCK-means algorithm was implemented as well. [102] Using the RMSD matrix as an input, the K-medoid algorithm is iterated, starting from two clusters and increasing number of clusters by one at each iteration. So the number of clusters at the ith iteration is $K_i = i + 1$. K_i-medoid clustering is repeated 100 times, then the result with the minimum mean of the squared distance of the clique within clusters (MSQw) is retained. The mean of the squared distance of the clique between clusters (MSQb) is calculated and used to calculate the simple moving average (SMA) of the ith iteration with a window of 10. Once the SMA begins decreasing, the clustering result having the maximum MSQb is returned.

We modified the algorithm to use K-medoid rather than K-means so that the algorithm was compatible with an RMSD matrix as input.

Ward/Dynamic Tree Cut

The previously described DTC algorithm was implemented and modified. [107] A previous publication demonstrated its application to conformational clustering. [102] A Ward dendrogram was obtained from the RMSD matrix, then subjected to the DTC algorithm.

[113] The DTC algorithm takes a tree or subtree and applies up to three thresholds. Cuts are considered significant only if the number of previous nodes above the threshold are greater than or equal to parameter $\tau \in \mathbb{N}$, for which we chose $\tau = 5$. The protocol is repeated on all subtrees until no new cuts are produced. Each cluster consists of leaves of the corresponding subtree. Representative structures were chosen by the medoid of each cluster. To simplify the algorithm, we did not perform the initial cut near the root of the tree, thereby beginning the adaptive tree cut protocol with a full dendrogram.

Fixed K Clusters on the Ward Dendrogram

We employed hierarchical clustering by the Ward dendrogram as a negative control to the adaptive clustering strategies. When applied to the 200 metabolite test set, the mean number of clusters across metabolites was determined to be 18.075. Thus we calculated the Ward dendrogram on the RMSD matrix and applied a threshold to produce 18 clusters.

3.2.4 Performance Evaluation

Our test sets included up to 1000 conformers per molecule sampled using the ETKDG algorithm, which were geometry optimized using various methods. Our premise is that different Hamiltonians and methods of minimization give different energy surfaces and geometries, so our test set contains conformers which have been biased toward their respective local minima. Therefore, our goal in clustering these conformers is to identify sets of conformers which would converge to similar geometries and energy values if fully minimized by a high quality QM method. If this was achieved by conformational clustering, we expect the similarity in geometry reflects similarity in energy values, and the cluster-wise variance in energy to be lower than if they were randomly sampled.

Correlation of Actual Energy to Local Weighted Estimation

AutoGraph clusters metabolite conformers based on the conformational graph generated. To justify clustering by the conformational graph for this test set, we assessed the geometric/energetic correlation in the graphs. We measured the single point energies of all conformers, which yields the actual measured energy. We also compute a local weighed

estimate of each node's energy by taking the average of neighbors' energy values weighted by incident edge weights. The Spearman correlation between actual and local estimated energy therefore provides a metric for the geometric/energetic correlation implicit in the conformational graph.

Formally, let the conformational graph $G = (V, E)$ consist of a set of vertices V connected by edges in set E. Let $v_i \in V$ and let its neighbors be $N_i = \{n_j | \{v_i, n_j\} \in E\} \subseteq V$. Note that $\{v_i, v_i\} \notin E$. Let $U(v)$ be the single point energy of the conformer assigned to vertex $v \in V$. Also, let $w(e)$ be the weight of edge $e \in E$. The actual energy of v_i is $U(v_i)$. The local estimated energy of v_i is given by,

$$\hat{U}(v_i) = \frac{\sum_{j=1}^{p} w(\{v_i, n_j\})U(n_j)}{\sum_{j=1}^{p} w(\{v_i, n_j\})}$$

The Spearman correlation coefficient was determined between U and $\hat{U}$, calculated in R. [114]

Cluster-wise Variance of Conformer Energy

Geometric similarity within clusters are achieved trivially by the objective function of the conformational clustering algorithm. We must instead consider the variance in the energies of the clustered conformers to evaluate the methodology. Energy as a metric informs us of the conformers' proximity in the PES and is independent of the clustering protocol, thus provides an independent metric for assessment. Note the AutoGraph algorithm considers conformer energy after partitioning conformers by clusters, thus its independence from assessment method is not compromised. For a distribution of values $X = \{x_i | i = 1, 2, \ldots, N\}$, the sample variance σ^2 is given by,

$$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N - 1}$$

in which $\bar{x} = \frac{\sum_i^N x_i}{N}$.

Because our conformers are organized by clusters, we can decompose the total variance

into variance between clusters and variance within clusters. Let c_j be the size of the jth cluster from 1 to C. Let μ_j be the mean energy of the jth cluster and μ be the mean energy of all conformers. The size of clusters are heterogeneous, thus we take a weighted mean to compute each terms as follows:

$$\sigma^2 = \sigma^2_{between} + \sigma^2_{within} = \sum_{j=1}^{C} \frac{c_j}{N}(\mu_j - \mu)^2 + \sum_{j=1}^{C} \frac{c_j}{N}\sigma_j^2$$

Suppose the conformers are clustered with no bias. Then $\sigma^2_{between}$ is negligible, thus we would expect $\sigma^2 \approx \sum_{j=1}^{C} \frac{c_j}{N}\sigma_j^2$. The converse of this statement is true, in which $\sigma^2 \neq \sum_{j=1}^{C} \frac{c_j}{N}\sigma_j^2$ implies a bias in clustering the energy. Our null hypothesis H_0 is $\sigma^2 = \sum_{j=1}^{C} \frac{c_j}{N}\sigma_j^2$. Our alternative hypothesis H_1 is $\sum_{j=1}^{C} \frac{c_j}{N}\sigma_j^2 < \sigma^2$. We detect a difference in variances using the F-test of equality of variances. [115]

To evaluate the energy variance among the 200 benchmark metabolites, we compute the variance in energy within clusters and variance across clusters for each of the 200 metabolites. We then apply the paired left-tailed Wilcoxon signed-rank test to determine the statistical significance of the difference between the two distributions for variance values. [116]

We impose a relatively stringent significance level α of 0.001 for the following reasons. First, the F-test is biased toward type-I error because $0 \leq \sigma^2_{between}$, thus we consider $\alpha_0 = 0.005$. Then we account for the inflated false positive rate due to multiple hypothesis testing on the same dataset over five algorithms using the Bonferroni correction. [117] By correcting for the number of null hypothesis significance tests, we obtain the final α of 0.001.

Within-Cluster RMSD

To obtain an intuition over the spread of geometries within clusters, we computed the mean RMSD for each cluster. The overall within-cluster RMSD was computed as a weighted mean of the mean RMSDs using the sizes of clusters, so that each conformer is equally weighed. The within-cluster RMSD values depend not only on the clustering algorithm, but also the conformational space of the metabolite. To standardize the comparison among metabolites, we subtract the global mean RMSD of the metabolite (mean RMSD without

considering clusters) from the overall within-cluster mean RMSD, and report the difference by algorithm and metabolite.

3.2.5 Case Studies

O-Succinyl-L-Homoserine

We clustered conformers generated for *O*-succinyl-L-homoserine (OSLH) to illustrate the use of the AutoGraph conformational clustering algorithm. Conformers were generated and refined in a previous study. [96] In summary, the MacroModel/ConfGen protocol (Schrödinger, Inc.) was used to generate 501 conformations unique to 0.1 Å in atomic RMSD. [118, 119]

This was followed by ANI-1ccx calculations in the gas phase. [95] Only conformers with no imaginary vibrational frequencies were retained, narrowing the conformers to 485. Calculations using ANI potentials were performed using the Atomic Simulation Environment (ASE) interface. [120]

Conformations were clustered by AutoGraph and the resultant graph was visualized using Gephi. [121] In addition, a charged structure was prepared using the PrepWizard tool (Schrödinger, Inc.). [122, 123] Single point energies were calculated using the Gaussian quantum chemistry software at the HF/6-31G(d) level of theory in the gas phase for both the neutral and charged conformers. [124]

ANI-1ccx optimized neutral OSLH conformers were further refined using Gaussian at the B3LYP/6-31G(d,p) level of theory in gas phase. Conformers were confirmed to be at local minima by vibrational analysis. The fully optimized geometries were subjected to clustering by AutoGraph.

Nicotinamide Adenine Dinucleotide

We also clustered conformers generated for nicotinamide adenine dinucleotide in the oxidized form (NAD+). We generated 1000 conformers from the SMILES of NAD+ using RDKit's implementation of the ETKDG algorithm, unique to 0.1 Å in atomic RMSD. [92, 125] Structures were optimized using the MMFF94 potential. [126] This was followed by the

Austin Model 1 (AM1) semiempirical method in the gas phase using the MOPAC software interfaced through ASE. [120, 127, 128] Only structures with the original topology were retained, filtering to 785 total conformers. Conformations were clustered by AutoGraph and the graphs were visualized using Gephi. [121] Single point energies were calculated using the Psi4 quantum chemistry package using the B3LYP/6-31G(d) level of theory in the gas phase. [129]

3.2.6 Benchmark Dataset

Metabolite Curation

Molecules were selected from the Human Metabolome Database (HMDB), consisting of 114184 metabolites. [130] Those localized to the cytosol, nucleus, or mitochondria were kept to yield 8249 molecules. We selected metabolites containing only elements which could be subjected to energy calculation by ANI-2x potentials (CHONSFCl) to yield 7547 molecules. [93] The number of rotatable bonds were calculated for each metabolite to remove trivial or unfeasible cases for conformation generation. Metabolites with rotatable bonds on the range of the 50th to 95th quantiles were retained, resulting in all metabolites having four to fourteen rotatable bonds. Out of the 3350 remaining metabolites, 200 representative structures were chosen using the following protocol. Morgan fingerprints were calculated for all molecules, using 2048 bits with a connectivity of three. A 3350×3350 matrix of Tanimoto distance between all fingerprints were calculated. A Ward dendrogram was calculated from the dissimilarity matrix and a threshold was applied to produce 200 clusters. Representative metabolites were selected from each cluster by having the greatest in-cluster degree. The metabolites are given in the appendix (Table A2).

Conformer Generation

Up to 1000 conformers were generated for each selected metabolite using RDKit's implementation of the ETKDG algorithm, discarding any structures with RMSD below 0.1 Å from any of the previous structures. [92, 125] All structures were optimized by the Merck Molecular Force Field 94 (MMFF94) in the gas phase. [126] Further, all structures were

optimized using ANI-2x potentials with the BFGS optimizer in the gas phase. [93] The final potential energies calculated with ANI-2x potentials were recorded.

3.3 Results and Discussion

3.3.1 Clustering *O*-succinyl-L-homoserine conformers by AutoGraph as an illustrative example

Among the strengths of using a graph to represent the conformers is graphs offer intuitive summaries of the relationships in the data, and an abundance of graph algorithms are available. AutoGraph was designed with the strategy to process the RMSD matrix into a readily interpretable form of a graph, then apply existing graph clustering protocols. We highlight the conformational graph of OSLH, for which each node represents exactly one conformer, and edge weights are proportional to the structural similarity determined by atomic RMSD between conformers (Figure 3.1).

Since the conformers are geometry optimized, we suspect the densely connected subgraphs represent basins in the PES, thus should be grouped in the same cluster. The AutoGraph protocol identified 28 clusters as shown. Nodes seem to have neighbors with similar energy, particularly in dense regions of the graph. The qualitative geometric/energetic correlation provides preliminary evidence for information regarding the PES is available implicitly in the conformational graph.

The superimposed conformers illustrate, while noisy, the overall molecular shape is similar within each cluster. The intuitive clusters on the graph appear to translate to qualitative conformational similarity for this system.

Unlike other clustering algorithms which take the RMSD matrix as the direct input, AutoGraph first processes the RMSD matrix into a graph representation. We assess whether the resulting conformational graph reasonably preserves the geometric/energetic similarities between conformers as expected for the PES. To do so, we measured the Spearman correlation coefficients ρ between the local estimated energy values on OSLH's conformational graph with their actual energy values, calculated by HF/6-31G(d) in the gas phase (Figure 3.2).

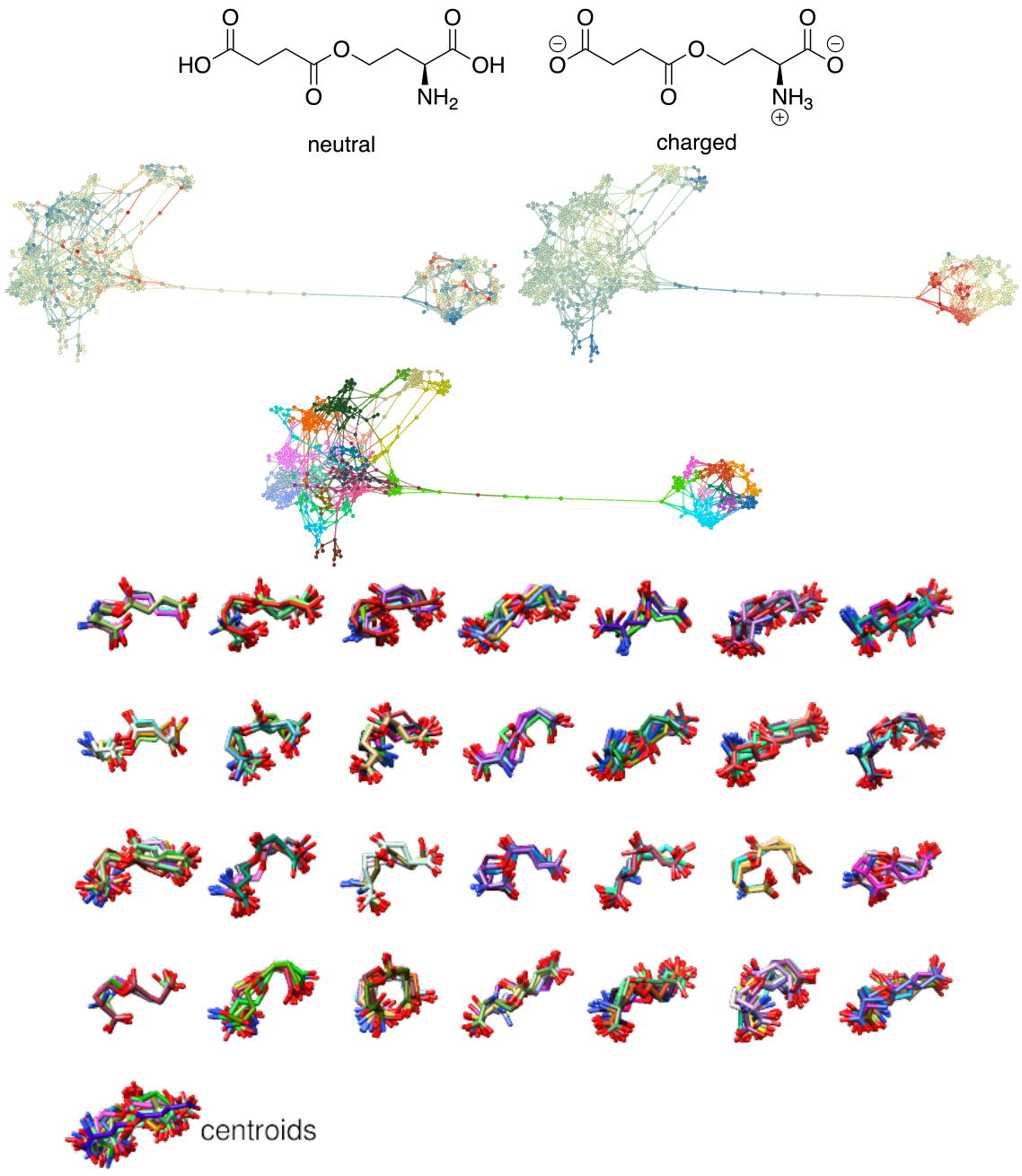

Figure 3.1: *O*-succinyl-L-homoserine (top) was subjected to conformer generation, geometry optimization, and clustering by AutoGraph. The conformational graphs colored by single point energy values for neutral (middle left) and charged (middle right) are shown using a gradient of blue (low energy) to red (high energy). The conformational graph colored by assigned cluster provide intuitive results (middle center). Conformers within each cluster were superimposed to their centroid (bottom).

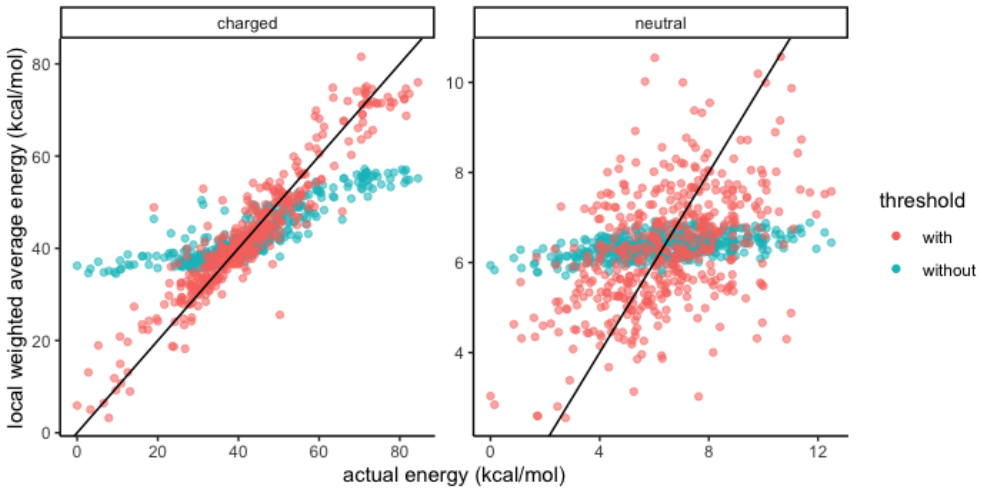

Figure 3.2: Locally weighted estimated relative energies was plotted against the measured relative single point energy values ($n = 485$). Graphs before and after filtering low weight edges were considered and Spearman correlation coefficients were calculated (charged: $\rho_{with} = 0.916, \rho_{without} = 0.852$; neutral: $\rho_{with} = 0.465, \rho_{without} = 0.457$). The line for $x = y$ is plotted.

There is a positive correlation for both conformational graphs, suggesting the monotonic relationship between geometry/energy is preserved even after kernelizing and filtering the RMSD matrix input. The local estimated energy appears more responsive to the actual energy when the adaptive threshold is applied. We also observe an improvement in the correlation. The improvement is particularly pronounced for the case of the charged system. We suspect the magnitude of Coulombic interactions in the gas phase is much greater than that of the noise such that we observe a strong relationship between the geometry and energy of the system. It is promising to observe a correlation in both the charged and neutral case even though their trends in relative energy differ. While the conformational graph represents purely geometric information, we observe we can infer energetic information because the refined conformations are biased to minima in the PES.

All ANI-1ccx optimized neutral OSLH were subjected to geometry optimization at B3LYP/6-31G(d,p) level of theory. The conformational graphs obtained by AutoGraph were visualized (Figure 3.3). In the one of the conformational graph, we indicate the conformers which were chosen as centroids by AutoGraph in the previous step for the ANI-1ccx

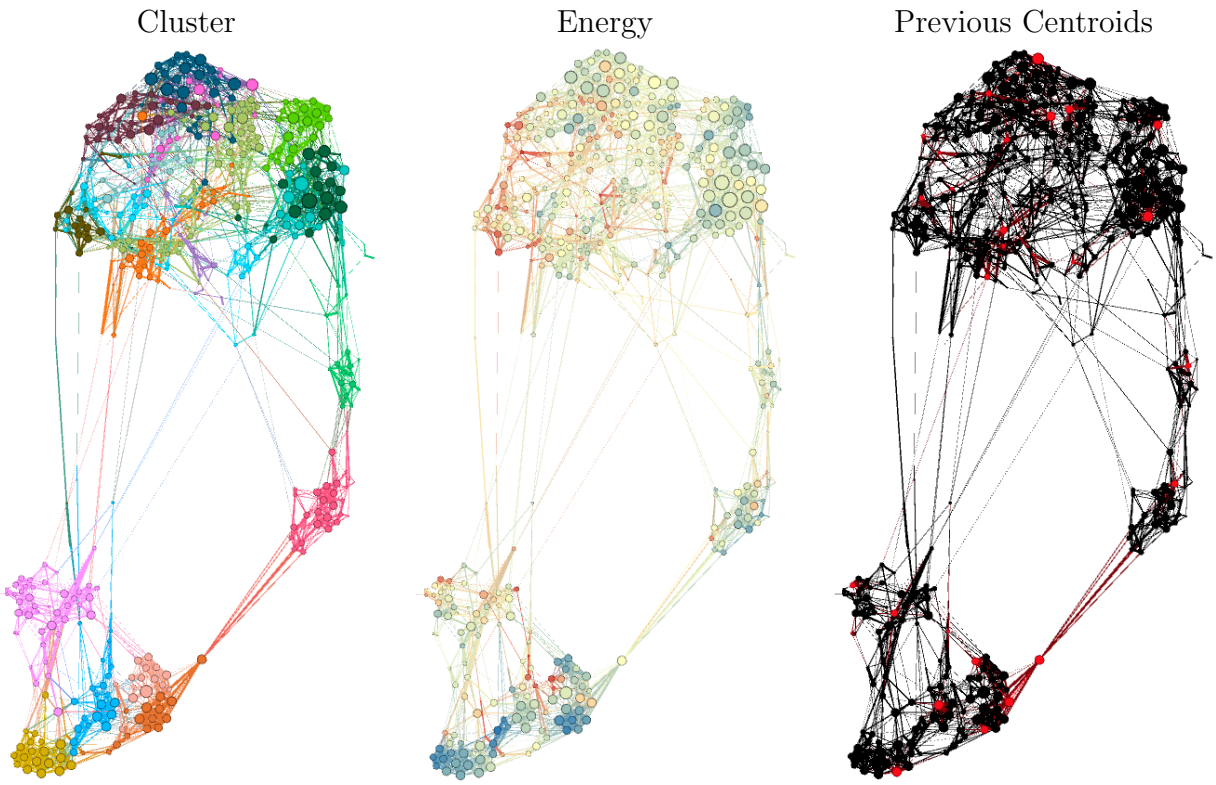

Figure 3.3: The conformational graph for B3LYP/6-31G(d,p) optimized neutral OSLH is colored by cluster assigned by AutoGraph, relative energy values from low (blue) to high (red), and centroids selected by AutoGraph from ANI-1ccx optimized OSLH conformers (red). Size of nodes are proportional to the weighted degree of each node.

optimized structures. We observe at least one structure from the centroids chosen from the ANI-1ccx optimized conformers appear in densely connected regions of the B3LYP/6-31G(d,p) optimized conformational graph. If we were to calculate an average property, we may select a representative conformer from each of the clusters of the B3LYP/6-31G(d,p) optimized conformational graph and take a Boltzmann average. We observe we can perform *ab initio* optimization on only the centroids selected from the ANI-1ccx optimized graph and obtain similar results as if we subjected all conformers to full geometry optimization. However, we can expedite the workflow in this case by subjecting only 28 starting conformers to the expensive geometry optimization rather than the full collection of 485 conformers.

3.3.2 Clustering nicotinamide adenine dinucleotide conformers by AutoGraph

We next examine the clustering result of NAD+ conformations. NAD+ is a well-known co-factor that mediates the redox currency in the cell. [131] Importantly, NAD+ contains two phosphorus atoms, which is an atom type not represented in the ANI-2x potential. We require an alternative protocol for refinement, and we can probe the robustness of the AutoGraph protocol to the method for conformer generation and refinement. After conformer generation and refinement with MMFF94, we optimized the NAD+ conformations using AM1 in gas phase. The 785 conformers which retained the original topology were subjected to conformational clustering by AutoGraph.

The AutoGraph protocol identified 31 clusters as shown (Figure 3.4). Overall, the graph appeared more globally connected than the OSLH example. The semiempirical QM method parameterized model 7 (PM7) has a lesser agreement to coupled cluster energy values when compared to ANI potentials, so the AM1 energies as a semiempirical method may have exhibited less convergence to local optima for NAD+ than ANI-2x did to OSLH. [94] The graph colored by single point energy values show an overall gradient, in which the densely connected region also appear to be low energy conformers. Meanwhile the higher energy side of the graph seems more sparsely connected. Superimposed conformers are also visually sound. While a positive correlation was observed between actual and local weighted energy estimates for conformational graphs before and after applying an edge weight threshold ($\rho_{with} = 0.845, \rho_{without} = 0.842$), no notable change in the correlation coefficient was observed (Figure A8). Because energetic information is inferred from the conformational graph, which only encodes geometric information explicitly, the success of the clustering results may depend on the convergence of optimized structures, which in turn depend on the accuracy of the energy calculation method. The throughput achieved by deep learning potentials like ANI potentials provide a unique opportunity for AutoGraph to interface between deep learning and *ab initio* methods in high throughput applications.

The cluster-wise variance in energy values were considered as an evaluation metric. The

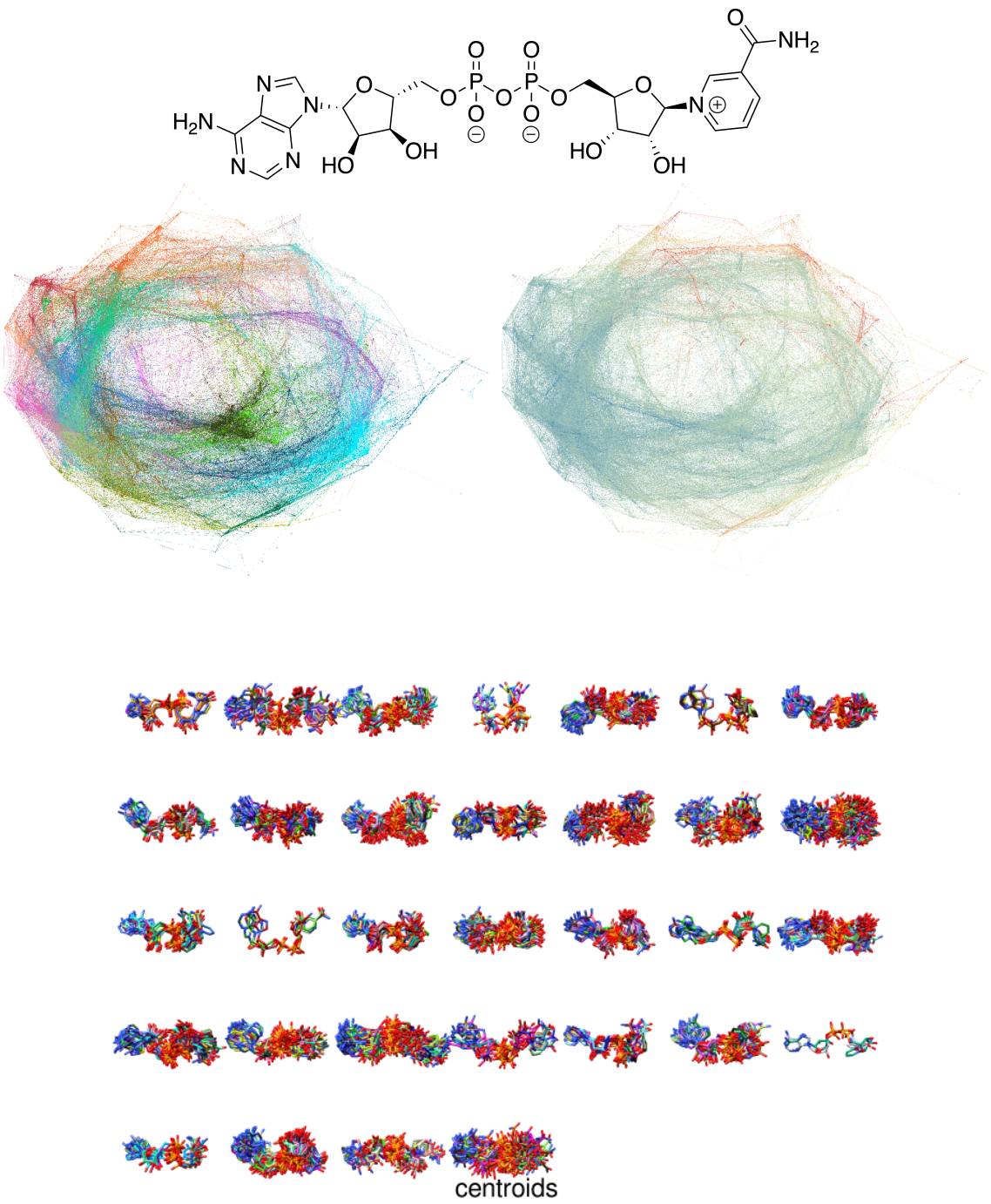

Figure 3.4: Nicotinamide adenine dinucelotide (top) was subjected to conformer generation, geometry optimization, and clustering by AutoGraph. The conformational graphs colored by cluster (middle left) and single point energy values (middle right) are shown using a gradient of blue (low energy) to red (high energy). Conformers within clusters were superimposed to their centroids (bottom).

Table 3.1: Variance in energy within clusters (σ^2_{within}) was compared against the variance among all conformers (σ^2) for OSLH and NAD+ individually. The F-test of equality of variances was used to assess the statistical significance of the difference in variances. The numbers of total conformers N and of clusters C are shown.

metabolite	N	C	σ^2	σ^2_{within}	p
$\text{OSLH}_{\text{neutral}}$	485	28	4.59	3.94	0.048
$\text{OSLH}_{\text{charged}}$	485	28	194.32	64.21	$< 0.001^*$
NAD+	785	31	105.67	49.15	$< 0.001^*$

Variances have units of $\left(\frac{kcal}{mol}\right)^2$. p-values were rounded to three decimal places. All other values were rounded to the second decimal place. Tests returning a p-value below 0.001 were considered significant, marked by $*$.

variance in energy within clusters was compared against the variance of energy among all conformers for OSLH and NAD+ individually (Table 3.1). We detect the within-cluster variance in energy is lesser than the overall variance for charged OSLH and for NAD+. The trend for neutral OSLH was not validated, likely due to its negligible effect size of $-0.65\left(\frac{kcal}{mol}\right)^2$ for the difference in variances.

While conformational clustering of OSLH and NAD+ both produced reasonable results, the results of this case studies are anecdotal. Validation of the AutoGraph protocol should be performed over a diverse data set chosen in a manner that minimizes bias. For this reason we constructed a benchmark conformation set for 200 metabolites chosen from the HMDB using a fairly automated protocol. [130]

3.3.3 Conformational graphs retain geometry/energy correlation for metabolites

The distribution of Spearman correlation coefficient obtained between actual and local weighted energy estimates on conformational graphs from all 200 metabolites were plotted (Figure 3.5). While the distributions exhibited a large range, the majority of ρ were positive, with median values of 0.36 before and 0.38 after applying the edge weight threshold. This indicates most graphs exhibited a positive monotonic relationship between actual and local estimated energy values to varying degrees. Proximity in the conformational graph therefore translates to similarity in energy. No significant enhancement was observed in applying

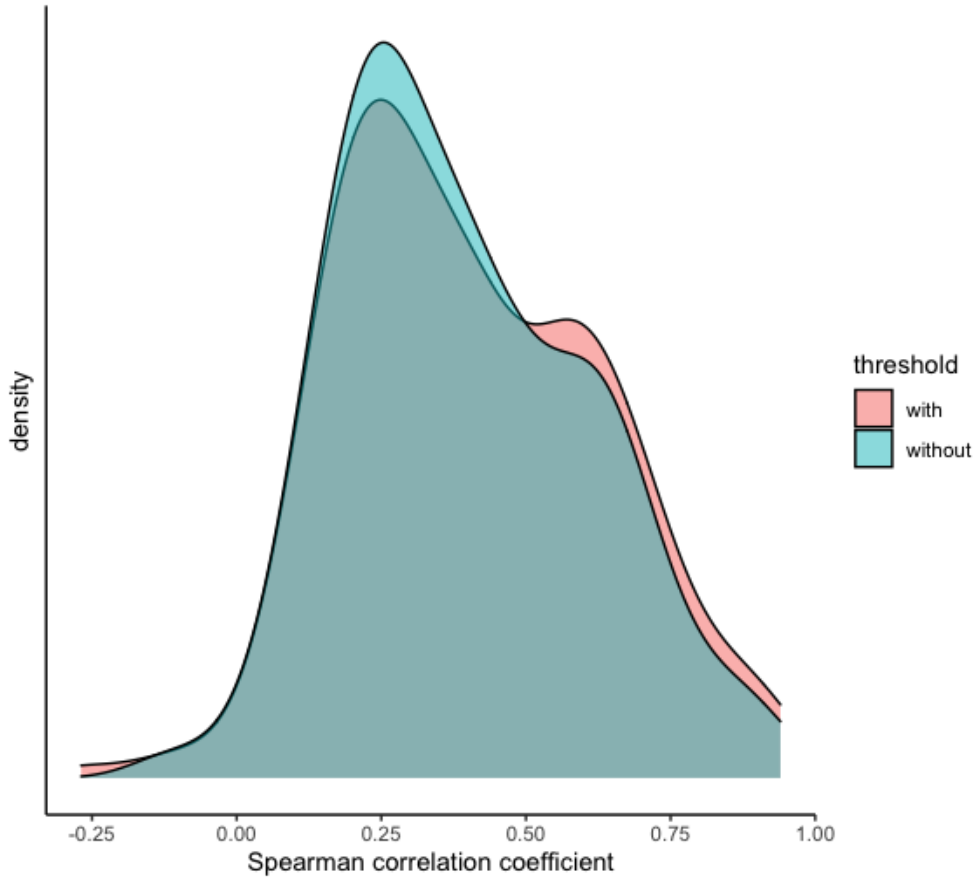

Figure 3.5: Distribution of Spearman correlation coefficients between actual and local estimated relative energy values by applying a with and without threshold value as part of the AutoGraph protocol to produce the conformational graphs ($n = 200$ for each distribution).

a threshold to the conformational graphs. We observe energetic information was inferred from the conformational graphs over a large, representative set of metabolites, therefore clustering by the conformational graphs may yield energetically informed partitions of the PES by thermally accessible local minima.

3.3.4 Autonomous clustering algorithms reduce mean within-cluster RMSD and cluster-wise variance in energy

We employed energy as a surrogate metric for the performance of clustering algorithms. Each dynamic clustering algorithm chooses its hyperparameters (e.g. number of clusters, threshold values) by optimizing its geometry based loss function. Thereby, we introduce the possibility of inflating the performance of a given algorithm had we chosen a geometry-

based assessment of the clustering algorithm. We report the difference between within-cluster mean RMSD and global mean RMSD for reference (Figure A9). However, we base our assessment on whether each clustering algorithm successfully clusters together partially optimized conformers with similar energy values, thus exhibit low within-cluster variance in energy.

The variance in energy within clusters were determined to be smaller than the total variance for charged OSLH and NAD+. We assessed the difference in cluster-wise and total variance in energy between the 200 benchmark structures across various algorithms (Table 3.2, Figure A10). We observe a significant decrease in cluster-wise variance in energy relative to the total variance for all dynamic clustering algorithms. The negative control failed to reject the null hypothesis ($H_0 : \sigma^2_{within} - \sigma^2 = 0$), highlighting the importance for automating the hyperparameter selection for the task of unsupervised clustering. Among the dynamic conformational clustering algorithms, AutoGraph exhibited the most negative mean and median discrepancy in variance of energy, $\sigma^2_{within} - \sigma^2$. The negative discrepancy in the variance of energy implies the cluster-wise variance is less than the overall variance. Conformational clustering algorithms successfully minimizing the variance in energy within the clusters indicates successful partitioning based on the biases imposed on sampled conformers during geometry optimization. Given this metric, AutoGraph appears to most successfully cluster conformers which we expect to converge to similar structures and energies upon further geometry optimization.

We considered the possibility in which a superficially low within-cluster variance in energy was achieved by forming only negligibly small clusters, thereby failing to generalize the clustering result. For each algorithm, The mean number of clusters detected per metabolite $\bar{C}$ was much smaller than the original number of conformers (≈ 1000), thus each algorithm appeared to successfully narrow down candidate conformers. We should note the effect size on $\sigma^2_{within} - \sigma^2$ is notably smaller than the results observed for charged OSLH and NAD+ (Table 3.1). While we consistently observe clustering by AutoGraph reduces the variance in energy

Table 3.2: The difference between variance in energy within clusters and variance among all conformers ($\sigma^2_{within} - \sigma^2$) for 200 representative metabolites were calculated ($df = 199$ per algorithm). The mean and median for each distributions were reported. A paired left-tailed Wilcoxon signed rank test was used to assess the statistical significance of the median discrepancy in variances in energy. The mean value of number of clusters detected $\bar{C}$ was also computed.

	AG	DTC	NC	RCK	fixed
mean	−8.18	−6.59	−2.57	−2.99	0.03
median	−1.09	−0.54	−0.38	−0.09	0.003
p	$< 0.001^*$	$< 0.001^*$	$< 0.001^*$	$< 0.001^*$	0.89
$\bar{C}$	18	18	32	4	18

Discrepancy in variances have units of $\left(\frac{kcal}{mol}\right)^2$. p-values were rounded to three decimal places. Variance discrepancies in energy were rounded to the second decimal place or to have at least one significant figure. $\bar{C}$ was rounded to the closest integer. Tests returning a p-value below 0.001 were considered significant, marked by $*$. Abbreviations: AutoGraph (AG), Ward/Dynamic Tree Cut (DTC), NMRCLUST (NC), Representative Conformer K-means (RCK), Ward hierarchical clustering with 18 clusters (fixed).

within clusters relative to the overall variance, the small effect size highlights AutoGraph as a tool to draw preliminary trends from a large number of metabolite conformations to reduce the search space for downstream processing and prediction, and not intended to make predictions itself.

In this study, we employed atomic RMSD as the distance metric because it is a simple, commonly used metric for structural similarity, applicable for geometry optimization results of small organic molecules. Various distance metrics may be more appropriate for other applications, such as atomic contacts or Fischer distance. [132] We implemented the AutoGraph protocol such that users can modify the dissimilarity/similarity metric used. This is achieved by populating the output directory with either the dissimilarity or affinity matrix (Figure A7). We welcome further assessment of the protocol applied to metrics other than atomic RMSD.

3.4 Conclusion

We presented use cases for an automated conformational clustering algorithm on OSLH and NAD+. Due to the throughput of ANI-2x potential and high degree of automation

of AutoGraph, we could generate, refine, and cluster the conformers for 200 representative metabolites. Further validation of the algorithm as a strategy to obtain an approximation of the underlying conformational ensemble is underway. The AutoGraph protocol can be integrated into computational workflows handling metabolite or other small-molecule conformations using a short Python script, or run as an interactive program with no coding required. We anticipate the application of AutoGraph will narrow down the search space in order to generate a representative conformational ensemble of collections of small-molecules.

Chapter 4: Rapid and Automated Ab Initio Metabolite Collisional Cross Section Prediction from SMILES Input

Kiyoto Aramis Tanemura, Susanta Das, Kenneth M. Merz Jr.

4.1 Introduction

Ion mobility spectrometry (IMS) is an analytical technique frequently coupled to mass spectrometry to separate analytes by size, shape, and charge for greater resolution. [19, 133] The collisional cross section (CCS) is a mobility measurement obtained by IMS, which provides a highly reproducible value for metabolite identification. [22] One untargeted metabolomics measurement can yield millions of signals, which is far greater than the databases of known CCS values. [134] High throughput metabolite identification therefore depends on both, the construction of more comprehensive CCS databases and the development of standard-free identification by in silico prediction of CCS values. [16]

Two main approaches are considered for computational prediction of CCS values: deduction from principle or inference from data. Ab initio methods model the gas phase conformational ensemble of charged species. The CCS can then be calculated for individual conformers and Boltzmann averaged. The in silico chemical library engine (ISiCLE) open source software is an example of this approach. [27] While the approach produces reliable CCS values justified by physical principles, molecular modeling involves computationally intensive simulation steps. In a complementary approach, data-driven methods parametrize a machine learning model on a database of molecules with known CCS values. AllCCS, DarkChem, DeepCCS are representative models of this group. [28, 31, 32] Such models are capable of rapid prediction of CCS values, however their accuracy depends on sufficient representation of similar structures in the training set; which is not guaranteed with the sparsity of available CCS values in relation to the vast number of possible metabolites.

To mitigate the computational bottleneck imposed by QM calculations, we augment an ab initio workflow for CCS prediction with an incremental refinement with the ANI deep

learning potentials.[93, 135] The developed workflow was validated against CCS values measured by traveling wave ion mobility-mass spectrometry (TWIMS) for a set of representative metabolites. [136] While the performance of the workflow was validated, execution of the workflow still presented a barrier.

The proper operation of computational chemistry software is required to execute this ab initio CCS prediction workflow, which is labor intensive and limits its use to expert computational chemists. The demand for concurrent development in annotation of experimental data and development in CCS prediction software further raises the barrier to accessibility in terms of expertise. Implementation of a highly automated workflow to execute advanced ab initio ensemble modeling and CCS calculation is therefore warranted for the metabolomics community who specialize in experimental or computational methods. Accessibility further benefits from the usage of free, open source software at every step of the computational pipeline.

The computational chemistry landscape is very complex with many packages and software suites vying for user attention. For all but the cognoscenti, this landscape is intimidating, making it hard for non-experts to work through a series of computational steps to get to a desired endpoint. Many decisions need to be made at each step in terms of the software to be used, its quality, reliability and accuracy. Thus, it is incumbent on the computational community to develop informatics infrastructure to support endpoints of import to experimental scientists. While the CCS prediction workflow is quite robust, its smooth implementation is a challenge even for expert users. Hence, the need for a workflow strategy that encapsulates all the tools needed to go from a simple molecular structure representation like SMILES to an accurate estimation of the CCS value.

The present work describes the informatics infrastructure and workflow implementation and does not focus on the computational details, which have been reported in detail elsewhere. [136] In light of this we describe the implementation of our CCS prediction workflow using the Snakemake workflow manager, which allows users to interact with a standardized

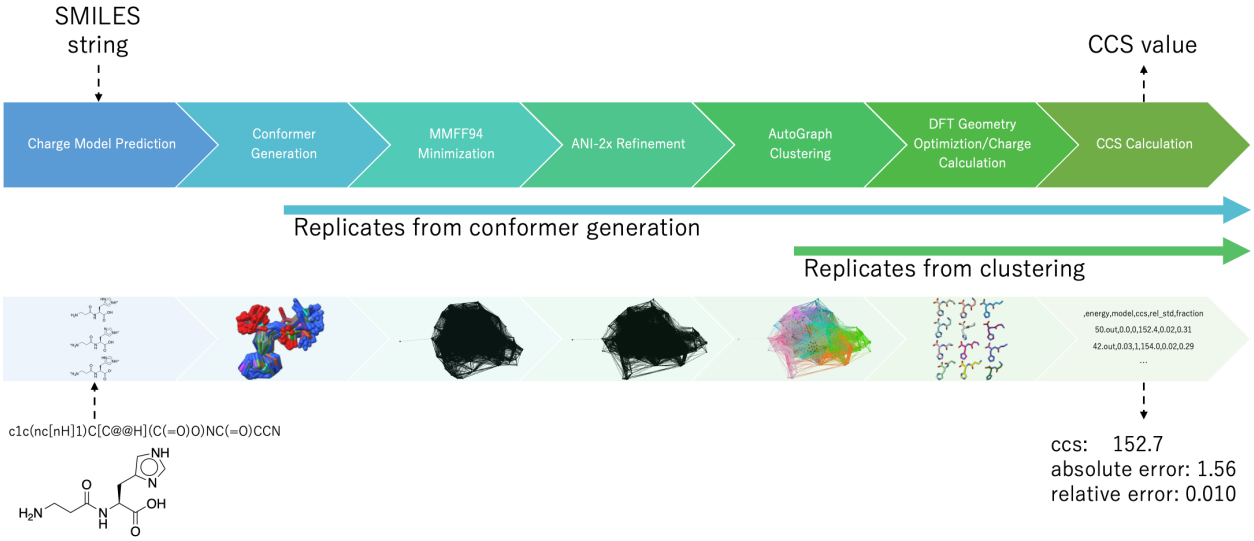

Figure 4.1: The present CCS prediction workflow was implemented in Snakemake such that a user can execute the workflow in an automated fashion, providing only the SMILES string as an input. We report the uncertainty introduced by stochastic steps in this workflow, the Conformer Generation and AutoGraph Clustering steps, by running ten replicates of independent CCS calculations from each step. Some visuals of the outputs are shown for the [M+H]+ adduct of L-carnosine to provide an intuition over each operation of the workflow. Briefly, three charge models are retained and up to 1000 conformers are generated for each. The conformational similarity graphs are shown after MMFF94 or ANI-2x refinement. We observe individual conformers become more modular as they approach minima in the potential energy surface. The AutoGraph clustering protocol then identifies clusters to maximize the modularity in the conformational similarity graph, and reports centroid conformers to be subject to computationally intensive DFT geometry optimization and CCS calculation. The CCS values are Boltzmann averaged and written to a result file accessible to the user.

interface to execute the multistep computation. [137] The high degree of automation allows users to provide only a simplified molecular-input line-entry system (SMILES) structure to predict CCS values for multiple protonated/deprotonated adducts and models. Furthermore, the automation allows for parallelized computation of CCS values on high performance computing (HPC) systems. Beyond describing the workflow implementation we also show its value by assessing the workflow by running replicate calculations starting from two stochastic steps in the workflow: conformer generation and clustering (Figure 4.1). In what follows we go over some of the details of the Snakemake implementation and how it can be used to assess the reliability of the workflow.

4.2 Description of the Implementation

The complex operations involving several open source software and unspecified numbers of files are automated using Snakemake. In Snakemake, computational workflows are programmed in a script which is titled the Snakefile. The Snakefile contains a list of target files and rules defined in blocks. The rules specify the input/output files, and the script to produce the output file from the input file. At execution, a directed acyclic graph (DAG) is computed based on the files present to plan the sequence of jobs to run. Snakemake executes the minimum jobs to produce all files specified in the target. [137]

Let us highlight as an example the following Snakefile, which produces output files by the HPCCS collisional cross section prediction software. [26] It defines the paths to the target files, the Boltzmann averaged CCS value saved to file *ccs.txt*, in a list. The operation takes coordinates and atomic charges from the QUICK QM output files and converts them to PQR format in rule *sp_out2pqr*. The CCS value is predicted from the PQR file using HPCCS, producing an output with extension .hpccs for the particular conformer. The HPCCS results are read from all conformers considered, Boltzmann averaged, and saved to *ccs.txt* through *rule model_results*.

```
# Target files are defined in a list by iterating through the metabolite ID,
# charge, and charge models considered
targets = []
for theID in ids:
    for thecharge in systems[theID]:
        if int(thecharge) not in args['consider_adducts_of_charges']:
            continue
        for themodel in systems[theID][thecharge]:
            targets.append('results/{}/{}/ensemble_fast/{}/ccs.txt'.format(theID, thecharge, themodel))

# The targets are passed to the rule titled all
```

```
rule all:
    input: targets

# Rules are defined in blocks. They specify input/output files, and the
   script to
# generate the output from the input

# write PQR file from QUICK single point output file
rule sp_out2pqr:
    input: "results/{label}/{charge}/ensemble_fast/{model}/{num}.out"
    output: "results/{label}/{charge}/ensemble_fast/{model}/{num}.pqr"
    shell:
           """
           src/sp_out2pqr.py {input}
           """

# Compute CCS value using HPCCS
rule hpccs:
    input: "results/{label}/{charge}/ensemble_fast/{model}/{num}.pqr"
    output: "results/{label}/{charge}/ensemble_fast/{model}/{num}.hpccs"
    params: hpccs = paths['HPCCS_path'],
            wdpath = paths['wd_path']
    shell: """
           cd {params.hpccs}
           ./hpccs {params.wdpath}{output} > {params.wdpath}{output}
           """

# Boltzmann average individual CCS values into final value
rule model_results:
    input: lambda wildcards: expand("results/{label}/{charge}/
   ensemble_fast/{model}/{num}.hpccs", label = wildcards.label, charge =
   wildcards.charge, model = wildcards.model, num = systems[wildcards.
   label][wildcards.charge][wildcards.model])
```

```
41      output: "results/{label}/{charge}/ensemble_fast/{model}/ccs.txt"
42      shell:
43              """
44              src/model_results_intermediate.py results/{wildcards.label}/{
    wildcards.charge}/ensemble_fast/{wildcards.model}/
45              """
```

Complex computational operations can be implemented in Snakemake by rules. Individual rules may have the forms of one-to-one, one-to-many, or many-to-one input files to output files. As long as the number of the files are specified, rules with many inputs or outputs can be encoded in one Snakefile. Workflows are allowed to diverge by using the same file as inputs for different rules, or converge by using files from different sources as input to one rule.

The number or identifiers of the intermediate files may not be defined at the beginning of the workflow, as is the case for our implementation which flexibly generates and selects conformers. This prevents the DAG from being computed for the entire workflow at the start. Therefore, we broke the workflow into convenient modules over which the DAG could be computed with no ambiguity, and the workflows encoded in these Snakefiles can be run sequentially inside a bash script. Snakemake therefore presents an opportunity to customize workflows using other methodologies or open source software in executing the CCS prediction workflow. We plan to explore this advantage further by fully automating the Portal for Open Computational Metabolomics Tools (POMICS) webpage (www.pomics.org). Currently, the web site serves as a contact page to compute predicted CCS values via this workflow; thereby requiring human intervention to initiate the workflow.

4.3 Description of Workflow

The computational details of the workflow is described in detail elsewhere. [136] Let us expand on the particular uses of software to automate the protocol, organized by sequential execution of workflows encoded in Snakefiles.

The input files are stored to the path *data/input.smi*. Arguments such as the charges

to consider are defined in *arguments.json*. Paths to various software is saved in *paths.json*.

4.3.1 Snakefile_make_model

The simplified molecular-input line-entry system (SMILES) structure is read. We employ Dimorphite-DL to protonate the functional groups present, thereby hypothesizing various protonation states. [138] Conformers are generated for each protonation model.

4.3.2 Snakemake_site_screen

The generated conformers are subjected to single point energy calculation in the gas phase using B3LYP/6-311G(d,p) through QUICK. [139–141] Models within 10 kcal/mol relative energy are retained as viable charge models.

While the implementation of the (de)protonation model is simple, it suffers from several drawbacks. Dimorphite-DL protonates common functional groups expected in the aqueous phase and not the gas-phase. Thus, the current implementation is prone to miss some viable gas-phase protonation states, such as protonated carbonyl groups or enolates. If the user has insight on how the metabolite might protonate, then they can append the SMILES string for the expected charged species in the charge model hypothesis file, model.smi. A more comprehensive protocol for metabolite protonation should be considered in future versions of the workflow.

4.3.3 Snakefile_conf_gen

Conformers are generated from the viable charge models using the ETKDG algorithm in RDKit. [92] The generated conformers are incrementally refined, first using the Merck Molecular Force Field 94 (MMFF94) implemented in RDKit to address distorted bonds and angles. [126]

4.3.4 Snakefile_AG

The refinement using MMFF94 is followed by refinement by the ANI2-x potential implemented in TorchANI. [93, 135] Following refinement, representative conformers are selected using AutoGraph conformational clustering. [142] Centroid structures are saved as the model of the conformational ensemble.

4.3.5 Snakefile_geom_opt

Centroid conformers are subjected to geometry optimization using B3LYP/6-31++G(d,p) in QUICK. The resulting geometries are a model for the gas-phase conformational ensemble for which we can use as inputs to predict average properties such as CCS values.

4.3.6 Snakefile_hpccs

Conformer-wise CCS values are predicted using High Performance Collision Cross Section (HPCCS) package. [26] The individual CCS values are Boltzmann averaged according to their optimized energies to yield the final CCS value, given the metabolite, charge, and charge model. The set of predicted CCS values are a metric that can be referenced against a measured CCS value to perform tasks such as metabolite identification or protonation model prediction.

4.4 Snakemake Enabled Workflow Extensions

Given the flexibility and reliability of the workflow, we can now innovate by using it in a number of ways by simply modifying the Snakefile to generate new endpoints. In this vein we highlight its use to create a "fast" workflow, assess workflow reliability, and the incorporation of NMR shift computation.

4.4.1 The Fast Workflow

The QM step is a computational bottleneck, and it may be helpful to obtain preliminary results even at a lower resolution. For this reason, we explored and now offer the fast workflow which performs a single point and charge calculation on the ANI2-x optimized centroid conformers rather than a full B3LYP/6-31++G(d,p) geometry optimization calculation. These conformers are subjected to CCS calculation and are Boltzmann weighted according to their B3LYP/6-31++G(d,p) single point energy, not the fully optimized energy.

The Snakemake implementation allowed us to readily create a representative benchmark to test predictions for positive and negative ion modes, using both the fast and standard workflows (Table A3). We used the *Snakefile_geom_opt* as a template, which includes the

geometry optimization step, and modified the source code such that the single point energy energy values were computed and used for Boltzman averaging the CCS values. Due to the modular fashion of the Snakemake workflow, the modification presented minimal coding effort.

The selection protocol of the benchmark set is further described in the appendix (sectino). The mean relative error (MRE) and Pearson correlation coefficient between predicted and experimental CCS values were calculated for each group (Table 4.1, Figure A11). If the benchmark contained only similar CCS values, then we may get a low MRE regardless of the input structure by arbitrarily choosing a value near the mean of the distribution of CCS values. Therefore as a negative control, we take the mean experimental CCS for each group and treat it as the predicted CCS for all compounds. This provides a best guess that does not distinguish input structure, thus having no predictive ability.

In both workflows, we observe improved accuracy and precision relative to the negative control, highlighting that the workflow is sensitive to input structures. We observe that the standard workflow exhibited a lower MRE than the fast workflow for both positive and negative ion modes. Meanwhile, the fast workflow achieves satisfactory performance for CCS prediction, using ANI2-x optimized structures in place of QM optimized structures. A high degree of linearity was reflected across all groups except for the control. The control included one prediction rather than a distribution, thereby was not applicable to calculate a correlation coefficient.

4.4.2 Assessment of Reproducibility

The workflow was executed to predict the CCS value of L-carnosine. To assess the reproducibility of stochastic steps in the workflow, all intermediate files generated subsequent to that step were deleted and allowed to rerun in ten replicate calculations. Conformer generation and conformational clustering are the stochastic steps involved in the workflow. The spread in CCS values produced thereafter represent the uncertainty introduced by the step and all subsequent operations (Table 4.2).

Table 4.1: The mean relative errors (MRE) of the *fast* and standard workflows over the benchmark set is reported, along with the negative control. The tests are grouped by charge and whether or not they are the negative control. The Pearson correlation coefficient (ρ) between the predicted and experimental CCS values are reported. The correlation coefficient cannot be calculated for the control group, which contains a singular value standing in for the prediction. All values reported to two significant figures.

workflow	charge	MRE ($\mu \pm 2\sigma$) (%)	ρ
fast	-1	3.4 ± 4.7	0.98
fast	1	3.1 ± 4.4	0.99
standard	-1	2.5 ± 4.7	0.97
standard	1	2.5 ± 4.5	0.98
control	-1	13 ± 13	-
control	1	13 ± 15	-

Table 4.2: The 95% confidence interval in CCS values obtained by ten replicate runs are reported for L-carnosine in positive/negative ion mode for viable models. The 95% confidence interval (CI) was calculated by two standard deviations, and reported as: absolute (relative). All values were rounded to one significant figure. Abbreviations: conformation generation (conf gen), conformational clustering (clustering).

charge	model	from	95% CI
1	0	conf gen	$0.9\mathring{A}^2(0.6\%)$
1	1	conf gen	$0.9\mathring{A}^2(0.6\%)$
1	2	conf gen	$1\mathring{A}^2(0.7\%)$
-1	0	conf gen	$2\mathring{A}^2(1\%)$
1	0	clustering	$0.4\mathring{A}^2(0.3\%)$
1	1	clustering	$0.4\mathring{A}^2(0.2\%)$
1	2	clustering	$0.2\mathring{A}^2(0.1\%)$
-1	0	clustering	$2\mathring{A}^2(1\%)$

In all cases considered, the 95% confidence interval was within 1% of the measured CCS value. Overall, the uncertainty introduced by using stochastic methods is marginal in this workflow, exhibiting high precision even with the use of stochastic algorithms for conformation generation and clustering.

4.4.3 NMR Calculation from Intermediate Files

One experiment is insufficient for tasks such as structure elucidation. Rather, we require convergent evidence to identify a metabolite from experimental spectra with confidence and integrity. In addition to measurements by IMS, nuclear magnetic resonance (NMR) spectroscopy on the sample can serve as another line of evidence for metabolite identification. It will be valuable to have predicted chemical shifts to compare against experimental values.

We illustrate that we can use intermediate files from modeling in the gas-phase ensemble to apply to an in silico NMR chemical shift prediction workflow. [96] A CCS prediction workflow was allowed to run to completion for O-succinyl-L-homoserine (OSLH). We then access a gas-phase optimized ensemble, located in the *results/OSLH/-1/ensemble/model0/* directory as *.xyz* files. We selected the –1 charge because it is the expected zwitterionic species in aqueous solution.

1H and 13C NMR chemical shifts were calculated as previously described, but using the structures in the gas phase neutral ensemble as input. [96] Briefly, the gas-phase optimized structures were subjected to geometry optimized using M06-2X/6-31+G(d,p), with the integrated equation formalism polarized continuum model (IEFPCM) with deuterium as solvent. 1H and 13C NMR shielding tensors were calculated using B3LYP/6-311G+(2d,p). Resulting shielding tensors were converted to NMR chemical shifts by linear regression. Finally, the shifts were Boltzmann averaged. The calculations were performed using Gaussian 16. [124]

The predicted chemical shifts were compared to measured values as well as those calculated by the original workflow (Table 4.3). We observe the mean absolute errors (MAE) to the experimental values are small in both predicted chemical shift values, however the au-

Table 4.3: Mean absolute errors (ppm, $\mu \pm 2\sigma$) of OSLH proton (upper right) and carbon (lower left) chemical shifts obtained by experiment, this workflow, and the expert supervised protocol.

		proton		
		experiment	workflow	expert
carbon	experiment	-	0.14 ± 0.19	0.31 ± 0.75
	workflow	1.1 ± 1.6	-	0.31 ± 0.70
	expert	2.4 ± 2.2	3.0 ± 3.0	-

tomated workflow requires minimal intervention. Because the MAE between chemical shifts predicted in each method are small, we consider the original expert supervised workflow reproduced. Thereby, we confirm the intermediate files generated in route to CCS prediction can be used for another application for which the conformational ensemble must be modeled.

4.5 Conclusion

The described implementation of the CCS prediction workflow uses freely available software using the SMILES structure as input. In addition to the standard workflow, we have a faster implementation which substitutes the computationally expensive QM geometry optimization with a single point energy calculation with a modest drop in accuracy. The reproducibility of the workflow has been validated for CCS calculation. Further, the intermediate files are useful for other applications which require a representative ensemble. The CCS prediction workflow is available at the following link: www.pomics.org

Chapter 5: Metabolite Collisional Cross Section and Uncertainty Prediction by Deep Bayesian Graph Convolutional Neural Network

Kiyoto Aramis Tanemura, Akhil Shajan, Susanta Das, Carter K. Asef, Facundo M. Fernández, Kenneth M. Merz Jr.

5.1 Introduction

Ab initio CCS prediction is a complementary approach to empirical CCS prediction. Quantitative structure property relation (QSPR) circumvents the complications involved in modeling and simulation, given sufficient data to parameterize a model. Meanwhile, the performance is dependent on the coverage and quality of data, and detailed structural information cannot be learned from QSPR alone. Therefore, while ML based CCS prediction models can rapidly generate *in silico* CCS libraries to be used in metabolite annotation, assessing the quality of these values remain a challenge.

We employ a graph convolutional deep Bayesian neural net to predict CCS directly from structure. [143] Graph representation provides a more intuitive representation of chemical structure than character embeddings of SMILES employed by DarkChem and DeepCCS. [31, 32] Additionally, uncertainty quantitation achieved by deep Bayesian neural net provides an informed method to handle the quality of values in the *in silico* library. The prior distribution can be estimated through introducing dropout to each parameter of the model. [144] The posterior distribution is then estimated through Bayesian inference by averaging the outcomes of Monte Carlo sampling T times.

Assuming a normal distribution for the output labels, the total uncertainty of the prediction is the predicted variance of the normal distribution, $\widehat{Var}(\hat{y}|x, \Theta)$. The total uncertainty is decomposed to the epistemic and aleatoric uncertainty. Epistemic uncertainty arises from the model or missing information, therefore the goal is to reduce epistemic uncertainty. Aleatoric uncertainty, on the other hand, is uncertainty which is inherent to the data. If the aleatoric uncertainty is constant across the domain, it is considered homoscedastic, and

can be represented by a singular uncertainty parameter. We assume the aleatoric uncertainty can vary depending on the metabolite, thereby model the aleatoric uncertainty to be heteroscedastic. The epistemic uncertainty is given by $\frac{1}{T}\sum_{t=1}^{T}(\hat{y}_t)^2 - \left(\frac{1}{T}\sum_{i=1}^{T}\hat{y}_t\right)^2$. The aleatoric uncertainty predicted by a neural network is given by, $\frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}_t^2$

CCS measurements have an uncertainty near 3% relative error. [136] Identical entries vary between experimental CCS databases by a mean relative error of 2.6%. Due to the relative sparsity of available CCS data relative to all possible metabolites, the uncertainty of predicted values likely are heterogeneous over predicted values. By use of the mean negative log likelihood (MNLL) as loss function, entries in the training set with lower predicted uncertainty can be weighted greater than those with higher predicted uncertainty. Meanwhile, uncertainty prediction provides a means to identify regions in the metabolite space with high uncertainty due to poor coverage.

Given the *ab initio* CCS prediction workflow is highly automated and independent of available data, we have the means to annotate regions of the metabolite space with less certainty of the CCS values. Combined with the uncertainty prediction achieved by the model, we can implement an active learning strategy to interplay between *ab initio* and deep learning CCS prediction methods.

5.2 Methods

5.2.1 Data

Three datasets were employed in this study: metabolite CCS, the human metabolome database (HMDB) set, and the test set. Each dataset contained the deposited SMILES, protonation state, and carrier gas. All but the HMDB set also contained experimental CCS values. The datasets were filtered by the following criteria:

- number of heavy atoms no more than 360
- contains no other elements than C/H/O/N/S/F/Cl
- adducts formed solely by protonation or deprotonation

Finally, entries with exact matches were dropped; entries with identical structures or experimental conditions were retained if the CCS values were not exact matches.

Metabolite Set

Experimental metabolite CCS values were obtained from various databases, including AllCCS, PNNL CCS Database, CCSBase, and Unified CCS Compendium. [28, 29, 145, 146] For training, entries with stereochemistry (stereogenic centers or olefin geometry) were identified. The dataset was appended by duplicate entries of the stereogenic compounds, but with the stereochemistry unspecified.

HMDB Set

All entries from the HMDB were obtained.[130] Structures present in the Metabolite Set were removed, thereby producing a collection of metabolites with no recorded CCS values. Monoprotonation and monodeprotonation adducts in nitrogen gas were hypothesized.

Test Set

To construct an independent test set, we sought to measure CCS values for structures not included in the training set. We curated candidates from a laboratory chemical inventory sheet. Chemicals containing no other elements than C/H/O/N/S/F/Cl were retained, yielding 268 candidates. Structures which were also present in the training set were removed, leaving 195 candidates. The Morgan fingerprints of the candidates and the training set were computed with connectivity of three and 2048 bits. The candidates were decorrelated by subjecting their Morgan fingerprints to Ward hierarchical clustering. A similarity threshold was applied to produce 50 clusters. For each cluster, the minimum Tanimoto distance of the candidates to all entries in the training set were calculated. The representative structure was chosen from each cluster as having the maximum *minimum distance* to structures in the training set. Up to 20 candidates were manually selected based on availability and feasibility, producing the final 16 candidates. The CCS values for these compounds were measured by a traveling wave IMS (TWIMS) (Table A4).

5.2.2 Featurization

Metabolites were featurized to the following arrays: the bond adjacency array, atomic descriptor array, experimental conditions array, and CCS labels. Let n be the number of entries in a dataset. The bond adjacency array A is a binary $n \times 360 \times 360$ array in which for each entry, the axes of the matrix correspond to the atom index of a molecule, and the presence of a covalent bond is indicated by '1'. The diagonal is also populated by '1' to convey self-similarity. The atomic descriptor matrix X is an $n \times 360 \times 16$ array encoding for atom type, chirality, olefin geometry, and number of hydrogens for each atom. The experimental conditions array is a $n \times 11$ array encoding for charge (-2 through $+6$), carrier gas (helium/nitrogen), and whether or not the entry is derived from experiment or theoretical prediction. Finally, the CCS labels array is a flat array of size n containing the known CCS values.

5.2.3 Graph Convolutional Deep Bayesian Neural Net

We based our graph convolutional deep Bayesian neural net on an existing model with gate-skip augmentation, and minor modifications (Figure 5.1). [143] The model consists of six graph convolution layers, the readout of a molecular descriptor, and two sets of three dense layers for the regression of the mean and variance of the predicted CCS values, respectively. Concrete dropout is applied after each layer. [144, 147] The parametric rectified linear unit (PReLU) activation function was used for all layers, except for the gate skipping mechanism. [148]

Graph Convolution with Gate Augmentation

Graph convolution takes the sum of the descriptors of each atom with its neighboring atoms, takes the dot product with trainable parameters, and passes through the PReLU activation function. Using the gate-skipping mechanism, a weighted average is computed between the present and previous convolution outputs, also using trainable parameters. The gate skipping mechanism employs the sigmoid activation function.

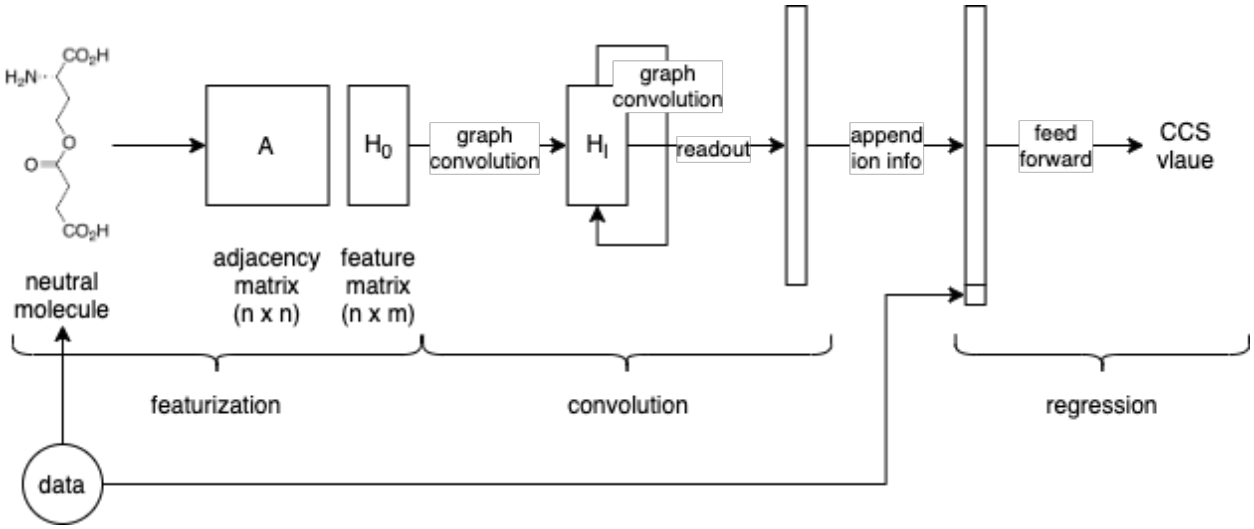

Figure 5.1: Graph convolutional deep Bayesian neural network with gate skipping augmentation, adapted from [143].

Readout of Molecular Descriptor

The atomic features passed through layers of graph convolution are subjected to a single layer perceptron (SLP). The outputs for each atom are summed to produce a molecular descriptor vector. The one-hot-encoded experimental conditions are appended. Feed Forward Regression of Mean and Variance

A standard multilayer perceptron (MLP) is used for the regression task of the mean and variance of CCS values. Each MLP outputs exactly one value, so the mean and the variance have dedicated MLP layers.

5.2.4 Training of the Model

The models were trained incrementally using different datasets and loss functions. In this study, sequential training refers to training the model in a deterministic manner before training it as a Bayesian model on the same training set. Transfer learning refers to training the model on the Peptide Set before the Metabolite Set.

Deterministic Training

The model is trained in a deterministic model by using the mean squared error (MSE) loss function, using the mean regression MLP and not the variance regression MLP. Let $\hat{y}$

be the predicted label and y be the true label. The MSE is given by,

$$MSE = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{5.1}$$

Bayesian Training

The model is trained as a Bayesian model by using the mean negative loss likelihood (MNLL) loss function. Let a Bayesian model have a set of parameters Θ, and compute the label from features, given by $f_\Theta(\mathbf{x}) = \hat{y}$. The negative log likelihood (NLL) is given by,

$$NLL = -\ln\left(P(y|\mathbf{x}, \Theta)\right) \tag{5.2}$$

in which $P(a|b)$ is the posterior probability of a, given the prior probability of b. Note that because $f_\Theta(\mathbf{x}) = \hat{y}, \hat{\sigma}^2$ for the Bayesian model f_Θ, $P(y|\mathbf{x}, \Theta) = P(y|\hat{y}, \hat{\sigma}^2)$. We parameterize the uncertainty as a Gaussian distribution, for which the likelihood is expressed as,

$$P(y|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right) \tag{5.3}$$

It follows the negative log likelihood is expressed as,

$$NLL = \frac{(y-\mu)^2}{2\sigma^2} + \frac{\ln \sigma^2}{2} \tag{5.4}$$

The mean log likelihood is the mean of the NLL for each entry;

$$MNLL = \frac{\sum_{i=1}^{n} NLL_i}{n} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(y_i - \hat{y}_i)^2}{2\hat{\sigma}_i^2} + \frac{\ln \hat{\sigma}_i^2}{2}\right) \tag{5.5}$$

5.2.5 CCS Mean and Variance Prediction

The mean and variance is determined using the Monte Carlo (MC) dropout method (100 samples). The aleatoric uncertainty is the mean of predicted variances for each entry.

The epistemic uncertainty is the variance in predicted means.

5.2.6 Performance Assessment

We average the performance metrics over five replicates of five-fold cross validation. The database contains structural and experimental redundancy because entries with different measured CCS values were retained. To prevent inflating the assessment metrics, we based the partitions on the metabolite SMILES, not indices of the training set.

For the CCS means prediction, we report the mean absolute error (MAE), root mean squared error (RMSE), and mean relative error (MRE) from the experimental CCS values.

For the CCS variance prediction, let us consider a metric we will refer to in this manuscript as the area under curve for variance (AUCV) to prevent confusion with the AUC of the receiver operating characteristic (ROC) curve. If the experimental CCS measurements were fairly sampled from the distribution, we then assume the z-score of each experimental CCS value to the predicted Gaussian distribution to also exhibit a Gaussian distribution. The predicted probability of sampling within a threshold absolute z-score is the relative area under a Gaussian probability distribution function within the z-score. The observed probability is the fraction of experimental CCS within the threshold absolute z-score. We plot the observed probability against the predicted probability to produce a curve from $(0.0, 0.0)$ to $(1.0, 1.0)$. If the two probabilities perfectly match, the AUCV of this curve has a value of 0.5. The AUCV is greater than 0.5 if the model is underconfident, predicting uncertainty values too large relative to the observed uncertainty. Meanwhile, an AUCV is below 0.5 if the model is overconfident, with predicted uncertainty smaller than the observed uncertainty.

5.2.7 Visualization of Molecular Encoding

To interpret the learned representation, the 512 feature molecular vector output of the readout layer was computed over all of the HMDB set. The vectors were subjected to principal component analysis (PCA) to a three dimensional embedding (ratio of variance explained: $PC0 : 87\%$, $PC1 : 7\%$, $PC2 : 4\%$). Various molecular descriptors were computed for each

structure using RDKit (*rdkit.Chem.rdMolDescriptors*). These well defined descriptors were overlaid on the PCA embedding of the molecular vectors.

5.2.8 Active Learning

A model trained on the entire metabolite training set predicts the mean and variance of CCS values over the HMDB set. The predictions are ordered by decreasing relative predicted uncertainty. The CCS values for the entries with high uncertainty are predicted using the *ab initio* workflow in batches of 100. The theoretical CCS values augment the training set in five iterations of five fold cross validation to evaluate the effect of theoretical augmentation.

The -1 and 1 charged adducts in nitrogen gas were hypothesized for the *ab initio* prediction. Only one charge model was considered for each metabolite/adduct pair. If no charged model was generated, the metabolite was removed by consideration. Therefore, while metabolites were considered in batches of 100, the number of theoretical CCS values augmented were not exactly 100.

5.3 Results and Discussion

5.3.1 Fast convergence of training is achieved by combining deterministic and Bayesian training

Performance metrics were tracked for the five replicates of five fold cross validation. The mean absolute error through training is shown (Figure 5.2). In *Bayesian* training, the model is trained with MNLL loss function for the whole 2400 epochs. In *Hybrid* training, the first 1200 epochs were trained with MSE loss by deterministic training, without training the variance prediction MLP. In the subsequent 1200 epochs, the Bayesian training was used to train both the mean and variance prediction networks. The decrease in MAE to below $10\mathring{A}$ occurs at above 1000 epochs for the Bayesian training, but is near 800 epochs for the Hybrid training. The final performance is similar between the two methods, but faster convergence was achieved by incremental training of the mean and variance networks using the Hybrid training.

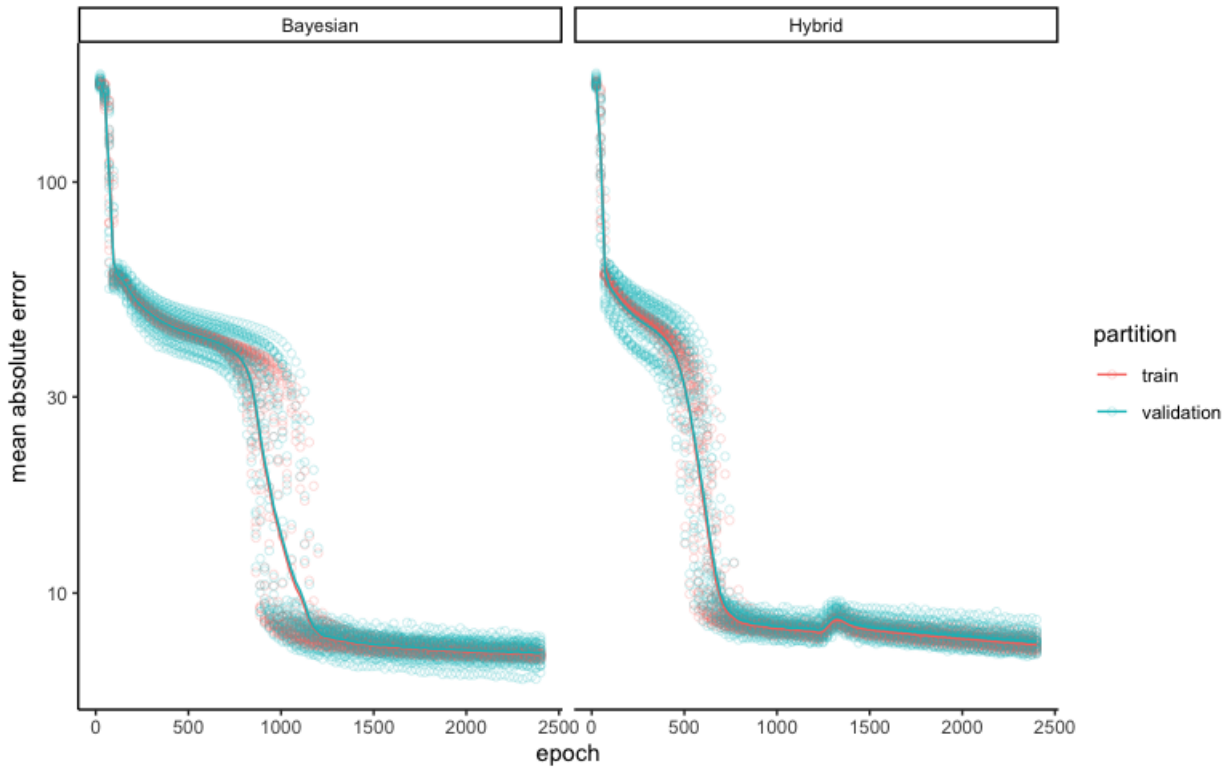

Figure 5.2: The mean absolute error values of five iterations of five fold cross validation, using only MNLL as the loss function (Bayesian) or using deterministic training followed by Bayesian training.

Table 5.1: Test set performance metrics by adduct charge and whether or not stereochemistry information was included. Performance metrics include mean absolute error (MAE), mean relative error (MRE), mean absolute z-score (MAZ), and mean predicted uncertainty (MPU). All values reported to two significant figures.

charge	stereochemistry	MAE (Å)	MRE (%)	MAZ	MPU (Å)
$+1$	yes	4.9	2.6	0.18	24
-1	yes	8.3	5.7	0.39	26
$+1/-1$	yes	6.5	4.1	0.28	25
$+1$	no	9.4	4.3	0.29	27
-1	no	10.	5.9	0.42	28
$+1/-1$	no	9.7	5.1	0.35	27

5.3.2 Missing stereochemistry is reflected in uncertainty of CCS prediction

The fully trained model was evaluated on the test set. To assess how the model treats systems with missing stereochemical information, molecules containing stereochemistry were assessed both with and without stereochemical information (Table 5.1). In all cases, the error increases upon loss of the stereochemical information. The loss of stereochemistry is accompanied by an increase in the mean predicted uncertainty. We illustrate the model is capable of handling compounds of missing CCS value, but reflect an increase in uncertainty.

5.3.3 Learned representation encodes information relevant for CCS prediction

The representation learned by graph convolution and readout were interpreted by visualization. We embedded the computed molecular vector to three dimensions by PCA and overlaid values computed by well defined molecular descriptors. Some plots which exhibited a recognizable pattern were selected and shown (Figure 5.3). These descriptors highlighted: Exact molecular weight, hydrogen bond acceptor count, ring count, and number of rotatable bonds; can affect the size and three dimensional shape of the molecule in the gas phase, thus is salient to CCS prediction. These features also vary in non-identical directions, thus are not completely correlated to one another. We observe the graph convolutional model is capable of embedding molecules from a graph structure to molecular vectors which preserves information such as size, flexibility, potential for intramolecular hydrogen bonding interaction.

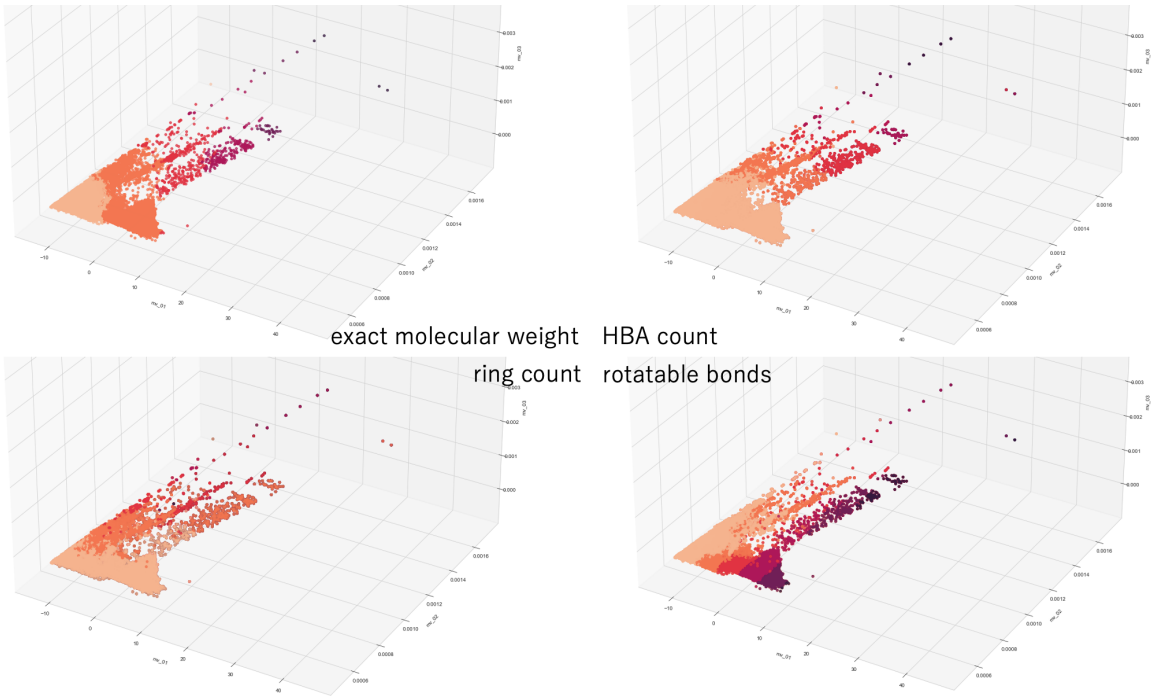

Figure 5.3: Exact molecular weight, hydrogen bond acceptor count, ring count, and number of rotatable bonds were overlaid on PCA embedding of molecular vectors of the HMDB set.

5.3.4 Augmentation of training set with theoretical CCS values improve performance

The HMDB set was sorted by decreasing predicted relative uncertainty to identify regions of the metabolite space. The CCS values of metabolites with the highest predicted relative uncertainty were predicted using the *ab inito* workflow in a high throughput and automated fashion. Theoretical values were added to training set of the model in batches of 100. The model was evaluated by five fold cross validation over solely the experimental CCS values; the theoretical values were added to the training set in all cases.

The five fold cross validation with no theoretical augmentation is shown at zero theoretical augmentation (Figure 5.4). As the number of theoretical CCS values are increased in the training set, the validation mean relative error decreases from 3.8% to 3.4%. The mean absolute error and root mean squared deviations both show a general decrease by the theoretical augmentations.

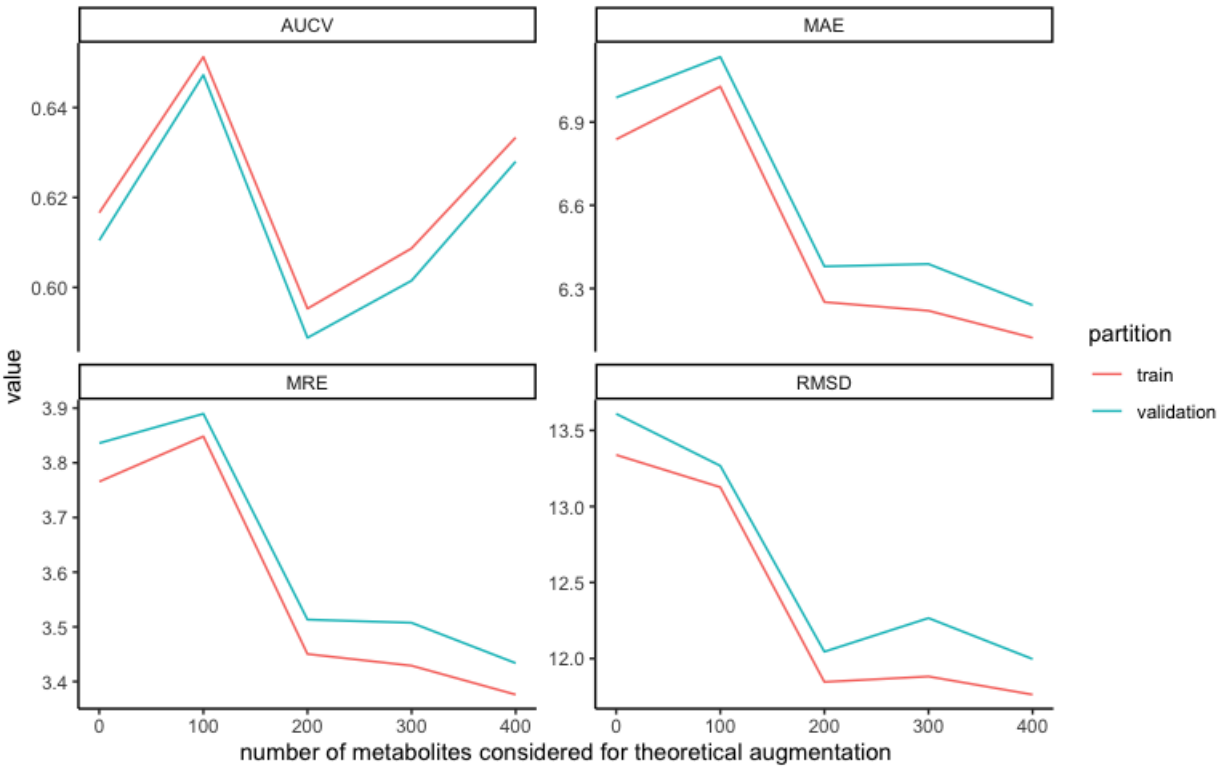

Figure 5.4: Performance metrics averaged over five iteration of five fold cross validation with augmentation of the training set with various numbers of theoretical CCS values. Abbreviations: area under curve of variance (AUCV, see methods), mean absolute error (MAE), mean relative error (MRE), root mean squared deviation (RMSD).

The area under curve of variance (AUCV) compares the expected and actual fraction of predictions within a Z-score threshold, and is used in this study to assess the predicted uncertainty. An AUCV at 0.5 shows an agreement of observed and predicted uncertainty. We observe AUCV values above 0.5, which shows the model is underconfident, exhibiting the model predicting uncertainty values greater than should be expected. There is no clear improvement of the validation AUCV by inclusion of theoretical CCS values to the training set.

The effect size may not exhibit a dramatic improvement, however the addition of merely 200 data points in a training set of 7331 points improved the performance. While the calculation of each theoretical CCS value is computationally nontrivial, involving multiple QM calculations per CCS value, the uncertainty quantitation by the deep Bayesian neural net provides a guided method to improve the theoretical annotation of CCS values.

5.4 Conclusion

The development of a gate augmented deep Bayesian graph convolutional neural net for CCS prediction offers an alternative representation of metabolite structure than existing deep learning based CCS prediction models. It presents the first instance of uncertainty quantitation of machine learning predicted CCS values. The uncertainty quantitation provides and informed method to perform active learning over the metabolite space, in combination with the automated, high throughput *ab initio* CCS prediction workflow. The active learning can be automated in future work to create a consinuously refining public database of in silico CCS values to aid in metabolite annotation.

Chapter 6: Thesis Contribution and Future Directions

In this body of research, I illustrate an increasing degree of automation achieved by use of machine learning in computational chemistry workflow development. Refinement of pairwise potential through the logistic regression classifier created a scoring function which evaluates protein-protein docking predictions with features which are consistent with the physical understanding of protein-protein interaction hot-spots. Inspired by manual operations which prevented automation of the NMR chemical shift prediction workflow, an unsupervised conformation clustering protocol was developed though a novel reformulation of RMSD distance to a similarity graph. The algorithm, AutoGraph, was implemented in the CCS prediction workflow using Snakemake, achieving a high degree of automation and throughput. Due to the automation achieved, the workflow could be evaluated in replicates, modified to a faster version, or extended to NMR chemical shift prediction. The workflow in tandem with a graph convolutional deep Bayesian neural net can perform active learning to predict CCS values of structures in regions of high uncertainty in the metabolite space.

This research contributes a collection of validated software toward the goal of comprehensive identification of metabolites from complex mixtures. The software we contribute are distinct from previously available CCS prediction software. In comparison to ISiCLE, our Snakemake-based workflow integrates an intermediate the ANI-2x deep learning potential followed by AutoGraph conformational clustering. The rapid refinement and adaptive narrowing of the conformational search space allows for *ab initio* CCS prediction on the timescale of minutes on the high performance computing cluster. Meanwhile, the deep learning based CCS prediction distinguishes itself from DeepCCS or DarkChem by uncertainty prediction. We provide the first example of uncertainty quantitation over deep learning based CCS prediction, thereby enabling the engineering of an active learning cycle between *ab initio* and deep learning methodologies.

Future work should focus on implementing software to automate the active learning

between the *ab initio* and deep learning predictions to create a continuously refining *in silico* database of CCS values. The workflows of the *ab initio* and deep learning based CCS prediction are individually automated, therefore an infrastructure to manage alternate execution of workflows can be developed with an orderly results output such that public release of the latest CCS values can be made available. Such values can be integrated to other databases such as the Human Metabolome Database. [14, 15] To maximize the quality of the training set, web scraping of existing experimental CCS databases can also be implemented. Subsequent publications should include the detailed assessment of the active learning of CCS values, the release of the continuously refining public CCS database, and the deployment of the library on realistic metabolite annotation.

The present work provides the foundation for other possible research directions. The AutoGraph conformational clustering was tailored specifically to expedite the NMR chemical shift/CCS prediction workflows. The application can be expanded to more challenging systems such as macromolecules or molecular dynamics trajectories by GPU-enabling the software through Tensorflow or PyTorch. The *ab initio* CCS prediction workflow was applied for metabolites, however preliminary evidence exhibits its utility to peptide and glycan CCS prediction as well. The scope of the workflow can be expanded by identifying and overcoming the specific challenges of the systems of interest. Finally, the deep learning based CCS workflow includes experimental conditions, unlike existing models, such that the predictions by a unified model can also be augmented by novel featurizers. Enhancement of the prediction performance can be pursued by convergent featurization of the molecule, possibly by including three dimensional features extracted by de Rham–Hodge analysis. [149]

In summary, I have developed though my PhD research a collection of software which was integrated to an active learning cycle for continuous refinement of metabolite CCS values. Subsequent studies should pursue the complete automation of the active learning cycle such that a comprehensive *in silico* CCS database can be downloaded at the time of analysis for the most reliable predicted CCS value for metabolite annotation. Meanwhile, the

developed software provide the foundation for alternative research directions. The body of work expands the capability of standard-free identification of metabolites by CCS prediction, thereby contributes to the effort of comprehensive structural elucidation in untargeted metabolomics experiments.

BIBLIOGRAPHY

(1) Gowda, G. N.; Zhang, S.; Gu, H.; Asiago, V.; Shanaiah, N.; Raftery, D. *Expert review of molecular diagnostics* **2008**, *8*, 617–633.

(2) Vinayavekhin, N.; Homan, E. A.; Saghatelian, A. *ACS chemical biology* **2010**, *5*, 91–103.

(3) Mamas, M.; Dunn, W. B.; Neyses, L.; Goodacre, R. *Archives of toxicology* **2011**, *85*, 5–17.

(4) Kaddurah-Daouk, R.; Kristal, B. S.; Weinshilboum, R. M., et al. *Annual review of pharmacology and toxicology* **2008**, *48*, 653–683.

(5) Toya, Y.; Shimizu, H. *Biotechnology advances* **2013**, *31*, 818–826.

(6) Dromms, R. A.; Styczynski, M. P. *Metabolites* **2012**, *2*, 1090–1122.

(7) Presnell, K. V.; Alper, H. S. *Biotechnology journal* **2019**, *14*, 1800416.

(8) Giani, A. M.; Gallo, G. R.; Gianfranceschi, L.; Formenti, G. *Computational and Structural Biotechnology Journal* **2020**, *18*, 9–19.

(9) Mirza, S. P.; Olivier, M. *Physiological genomics* **2008**, *33*, 3–11.

(10) Bohacek, R. S.; McMartin, C.; Guida, W. C. *Medicinal research reviews* **1996**, *16*, 3–50.

(11) Polishchuk, P. G.; Madzhidov, T. I.; Varnek, A. *Journal of computer-aided molecular design* **2013**, *27*, 675–679.

(12) Yanes, O.; Tautenhahn, R.; Patti, G. J.; Siuzdak, G. *Analytical chemistry* **2011**, *83*, 2152–2161.

(13) Naser, F. J.; Mahieu, N. G.; Wang, L.; Spalding, J. L.; Johnson, S. L.; Patti, G. J. *Analytical and bioanalytical chemistry* **2018**, *410*, 1287–1297.

(14) Wishart, D. S.; Tzur, D.; Knox, C.; Eisner, R.; Guo, A. C.; Young, N.; Cheng, D.; Jewell, K.; Arndt, D.; Sawhney, S., et al. *Nucleic acids research* **2007**, *35*, D521–D526.

(15) Wishart, D. S.; Guo, A.; Oler, E.; Wang, F.; Anjum, A.; Peters, H.; Dizon, R.; Sayeeda, Z.; Tian, S.; Lee, B. L., et al. *Nucleic Acids Research* **2022**, *50*, D622–D631.

(16) Borges, R. M.; Colby, S. M.; Das, S.; Edison, A. S.; Fiehn, O.; Kind, T.; Lee, J.; Merrill, A. T.; Merz Jr, K. M.; Metz, T. O., et al. *Chemical reviews* **2021**, *121*, 5633–5670.

(17) Markley, J. L.; Brüschweiler, R.; Edison, A. S.; Eghbalnia, H. R.; Powers, R.; Raftery, D.; Wishart, D. S. *Current opinion in biotechnology* **2017**, *43*, 34–40.

(18) Sindelar, M.; Patti, G. J. *Journal of the American Chemical Society* **2020**, *142*, 9097–9105.

(19) Gabelica, V.; Marklund, E. *Current opinion in chemical biology* **2018**, *42*, 51–59.

(20) Revercomb, H.; Mason, E. A. *Analytical chemistry* **1975**, *47*, 970–983.

(21) May, J. C.; Goodwin, C. R.; Lareau, N. M.; Leaptrot, K. L.; Morris, C. B.; Kurulugama, R. T.; Mordehai, A.; Klein, C.; Barry, W.; Darland, E., et al. *Analytical chemistry* **2014**, *86*, 2107–2116.

(22) Paglia, G.; Williams, J. P.; Menikarachchi, L.; Thompson, J. W.; Tyldesley-Worster, R.; Halldórsson, S.; Rolfsson, O.; Moseley, A.; Grant, D.; Langridge, J., et al. *Analytical chemistry* **2014**, *86*, 3985–3993.

(23) Mesleh, M.; Hunter, J.; Shvartsburg, A.; Schatz, G. C.; Jarrold, M. *The Journal of Physical Chemistry* **1996**, *100*, 16082–16086.

(24) Shvartsburg, A. A.; Jarrold, M. F. *Chemical physics letters* **1996**, *261*, 86–91.

(25) Marklund, E. G.; Degiacomi, M. T.; Robinson, C. V.; Baldwin, A. J.; Benesch, J. L. *Structure* **2015**, *23*, 791–799.

(26) Zanotto, L.; Heerdt, G.; Souza, P. C.; Araujo, G.; Skaf, M. S. High performance collision cross section calculation—HPCCS, 2018.

(27) Colby, S. M.; Thomas, D. G.; Nuñez, J. R.; Baxter, D. J.; Glaesemann, K. R.; Brown, J. M.; Pirrung, M. A.; Govind, N.; Teeguarden, J. G.; Metz, T. O., et al. *Analytical chemistry* **2019**, *91*, 4346–4356.

(28) Zhou, Z.; Luo, M.; Chen, X.; Yin, Y.; Xiong, X.; Wang, R.; Zhu, Z.-J. *Nature communications* **2020**, *11*, 1–13.

(29) Ross, D. H.; Cho, J. H.; Xu, L. *Analytical chemistry* **2020**, *92*, 4548–4557.

(30) Nguyen, K. T.; Blum, L. C.; Van Deursen, R.; Reymond, J.-L. *ChemMedChem: Chemistry Enabling Drug Discovery* **2009**, *4*, 1803–1805.

(31) Plante, P.-L.; Francovic-Fontaine, É.; May, J. C.; McLean, J. A.; Baker, E. S.; Laviolette, F.; Marchand, M.; Corbeil, J. *Analytical chemistry* **2019**, *91*, 5191–5199.

(32) Colby, S. M.; Nuñez, J. R.; Hodas, N. O.; Corley, C. D.; Renslow, R. R. *Analytical chemistry* **2019**, *92*, 1720–1729.

(33) Samuel, A. L. *IBM Journal of Research and Development* **1959**, *3*, 210–229.

(34) Tolle, K. M.; Tansley, D. S. W.; Hey, A. J. *Proceedings of the IEEE* **2011**, *99*, 1334–1337.

(35) Cybenko, G. *Mathematics of control, signals and systems* **1989**, *2*, 303–314.

(36) Laraia, L.; McKenzie, G.; Spring, D. R.; Venkitaraman, A. R.; Huggins, D. J. *Chemistry & biology* **2015**, *22*, 689–703.

(37) Bakail, M.; Ochsenbein, F. *Comptes Rendus Chimie* **2016**, *19*, 19–27.

(38) Jaeger, A. M.; Whitesell, L. *Annual Review of Cancer Biology* **2019**, *3*, 275–297.

(39) Neckers, L.; Blagg, B.; Haystead, T.; Trepel, J. B.; Whitesell, L.; Picard, D. *Cell Stress and Chaperones* **2018**, *23*, 467–482.

(40) Sattin, S.; Tao, J.; Vettoretti, G.; Moroni, E.; Pennati, M.; Lopergolo, A.; Morelli, L.; Bugatti, A.; Zuehlke, A.; Moses, M., et al. *Chemistry–A European Journal* **2015**, *21*, 13598–13608.

(41) Paladino, A.; Woodford, M. R.; Backe, S. J.; Sager, R. A.; Kancherla, P.; Daneshvar, M. A.; Chen, V. Z.; Bourboulia, D.; Ahanin, E. F.; Prodromou, C., et al. *Chemistry–A European Journal* **2020**, *26*, 9459–9465.

(42) Dobson, C. M. *Annual review of biochemistry* **2019**, *88*.

(43) Vajda, S.; Hall, D. R.; Kozakov, D. *Proteins: Structure, Function, and Bioinformatics* **2013**, *81*, 1874–1884.

(44) Pierce, B. G.; Hourai, Y.; Weng, Z. *PloS one* **2011**, *6*, e24657.

(45) Gabb, H. A.; Jackson, R. M.; Sternberg, M. J. *Journal of molecular biology* **1997**, *272*, 106–120.

(46) Tovchigrechko, A.; Vakser, I. A. *Nucleic acids research* **2006**, *34*, W310–W314.

(47) De Vries, S.; Zacharias, M. *Proteins: Structure, Function, and Bioinformatics* **2013**, *81*, 2167–2174.

(48) Moal, I. H.; Bates, P. A. *International journal of molecular sciences* **2010**, *11*, 3623–3648.

(49) Schindler, C. E.; de Vries, S. J.; Zacharias, M. *Proteins: Structure, Function, and Bioinformatics* **2015**, *83*, 248–258.

(50) Pierce, B.; Weng, Z. *Proteins: Structure, Function, and Bioinformatics* **2007**, *67*, 1078–1086.

(51) Fiorucci, S.; Zacharias, M. *Proteins: Structure, Function, and Bioinformatics* **2010**, *78*, 3131–3139.

(52) Camacho, C. J.; Zhang, C. *Bioinformatics* **2005**, *21*, 2534–2536.

(53) Andrusier, N.; Nussinov, R.; Wolfson, H. J. *Proteins: Structure, Function, and Bioinformatics* **2007**, *69*, 139–159.

(54) Park, T.; Baek, M.; Lee, H.; Seok, C. *Journal of computational chemistry* **2019**.

(55) Feng, T.; Chen, F.; Kang, Y.; Sun, H.; Liu, H.; Li, D.; Zhu, F.; Hou, T. *Journal of cheminformatics* **2017**, *9*, 66.

(56) De Vries, S. J.; van Dijk, A. D.; Krzeminski, M.; van Dijk, M.; Thureau, A.; Hsu, V.; Wassenaar, T.; Bonvin, A. M. *Proteins: structure, function, and bioinformatics* **2007**, *69*, 726–733.

(57) Comeau, S. R.; Gatchell, D. W.; Vajda, S.; Camacho, C. J. *Bioinformatics* **2004**, *20*, 45–50.

(58) Andreani, J.; Faure, G.; Guerois, R. *Bioinformatics* **2013**, *29*, 1742–1749.

(59) Khashan, R.; Zheng, W.; Tropsha, A. *Proteins: Structure, Function, and Bioinformatics* **2012**, *80*, 2207–2217.

(60) Zhou, H.; Zhou, Y. *Protein science* **2002**, *11*, 2714–2726.

(61) Yang, Y.; Zhou, Y. *Proteins: Structure, Function, and Bioinformatics* **2008**, *72*, 793–803.

(62) Moreira, I. S.; Fernandes, P. A.; Ramos, M. J. *Proteins: Structure, Function, and Bioinformatics* **2007**, *68*, 803–812.

(63) Scarabelli, G.; Morra, G.; Colombo, G. *Biophysical journal* **2010**, *98*, 1966–1975.

(64) Mihalek, I.; Reš, I.; Lichtarge, O. *Journal of molecular biology* **2004**, *336*, 1265–1282.

(65) Marchetti, F.; Capelli, R.; Rizzato, F.; Laio, A.; Colombo, G. *The Journal of Physical Chemistry Letters* **2019**, *10*, 1489–1497.

(66) Basu, S.; Wallner, B. *Bioinformatics* **2016**, *32*, i262–i270.

(67) Geng, C.; Jung, Y.; Renaud, N.; Honavar, V.; Bonvin, A. M.; Xue, L. C. *BioRxiv* **2018**, 498584.

(68) Pfeiffenberger, E.; Chaleil, R. A.; Moal, I. H.; Bates, P. A. *Proteins: Structure, Function, and Bioinformatics* **2017**, *85*, 528–543.

(69) Wang, X.; Terashi, G.; Christoffer, C. W.; Zhu, M.; Kihara, D. *Bioinformatics* **2020**, *36*, 2113–2118.

(70) Pei, J.; Zheng, Z.; Merz Jr, K. M. *Journal of chemical information and modeling* **2019**.

(71) Pei, J.; Zheng, Z.; Kim, H.; Song, L. F.; Walworth, S.; Merz, M. R.; Merz, K. M. *Journal of Chemical Information and Modeling* **2019**.

(72) Zheng, Z.; Merz Jr, K. M. *Journal of chemical information and modeling* **2013**, *53*, 1073–1083.

(73) Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. *Journal of computational chemistry* **2004**, *25*, 1157–1174.

(74) Han, L.; Yang, Q.; Liu, Z.; Li, Y.; Wang, R. *Future medicinal chemistry* **2018**, *10*, 1555–1574.

(75) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V., et al. *Journal of machine learning research* **2011**, *12*, 2825–2830.

(76) Méndez, R.; Leplae, R.; De Maria, L.; Wodak, S. J. *Proteins: Structure, Function, and Bioinformatics* **2003**, *52*, 51–67.

(77) May, A.; Zacharias, M. *Proteins: Structure, Function, and Bioinformatics* **2007**, *69*, 774–780.

(78) Zhang, Q.; Feng, T.; Xu, L.; Sun, H.; Pan, P.; Li, Y.; Li, D.; Hou, T. *Current drug targets* **2016**, *17*, 1586–1594.

(79) Vreven, T.; Moal, I. H.; Vangone, A.; Pierce, B. G.; Kastritis, P. L.; Torchala, M.; Chaleil, R.; Jiménez-Garcıa, B.; Bates, P. A.; Fernandez-Recio, J., et al. *Journal of molecular biology* **2015**, *427*, 3031–3041.

(80) De Vries, S. J.; Schindler, C. E.; de Beauchêne, I. C.; Zacharias, M. *Biophysical journal* **2015**, *108*, 462–465.

(81) Chen, R.; Mintseris, J.; Janin, J.; Weng, Z. *Proteins: Structure, Function, and Bioinformatics* **2003**, *52*, 88–91.

(82) Kumar, S.; Nussinov, R. *ChemBioChem* **2002**, *3*, 604–617.

(83) Li, Y.; Zhang, X.; Cao, D. *Scientific reports* **2013**, *3*, 3271.

(84) Kubo, R.; McQuarrie, D. A. *Phys. Today* **1965**, *18*, 74–75.

(85) Amaro, R. E.; Baudry, J.; Chodera, J.; Demir, Ö.; McCammon, J. A.; Miao, Y.; Smith, J. C. *Biophys. J.* **2018**, *114*, 2271–2278.

(86) Verma, J.; Khedkar, V.; Coutinho, E. *Curr. Top. Med. Chem.* **2010**, *10*, 95–115.

(87) Husic, B. E.; Pande, V. S. *J. Am. Chem. Soc.* **2018**, *140*, 2386–2396.

(88) Hawkins, P. C. *J. Chem. Inf. Model.* **2017**, *57*, 1747–1756.

(89) Pracht, P.; Bohle, F.; Grimme, S. *Phys. Chem. Chem. Phys.* **2020**, *22*, 7169–7192.

(90) Wilson, S. R.; Cui, W.; Moskowitz, J. W.; Schmidt, K. E. *J. Comput. Chem.* **1991**, *12*, 342–349.

(91) Hawkins, P. C.; Nicholls, A. *J. Chem. Inf. Model.* **2012**, *52*, 2919–2936.

(92) Riniker, S.; Landrum, G. A. *J. Chem. Inf. Model.* **2015**, *55*, 2562–2574.

(93) Devereux, C.; Smith, J. S.; Davis, K. K.; Barros, K.; Zubatyuk, R.; Isayev, O.; Roitberg, A. E. *J. Chem. Theory Comput.* **2020**, *16*, 4192–4202.

(94) Folmsbee, D.; Hutchison, G. *Int. J. Quantum Chem.* **2020**, *121*, e26381.

(95) Smith, J. S.; Nebgen, B. T.; Zubatyuk, R.; Lubbers, N.; Devereux, C.; Barros, K.; Tretiak, S.; Isayev, O.; Roitberg, A. E. *Nat. Commun.* **2019**, *10*, 1–8.

(96) Das, S.; Edison, A. S.; Merz, K. M. *Anal. Chem.* **2020**, *92*, 10412–10419.

(97) Xu, D.; Tian, Y. *Ann. Data. Sci.* **2015**, *2*, 165–193.

(98) Solorio-Fernández, S.; Carrasco-Ochoa, J. A.; Martínez-Trinidad, J. F. *Artif. Intell. Rev.* **2019**, *53*, 907–948.

(99) Shao, J.; Tanner, S. W.; Thompson, N.; Cheatham, T. E. *J. Chem. Theory Comput.* **2007**, *3*, 2312–2334.

(100) Yongye, A. B.; Bender, A.; Martínez-Mayorga, K. *J. Comput. Aided Mol. Des.* **2010**, *24*, 675–686.

(101) Li, Y.; Dong, Z. *J. Chem. Inf. Model.* **2016**, *56*, 1205–1215.

(102) Kim, H.; Jang, C.; Yadav, D. K.; Kim, M.-h. *J. Cheminform.* **2017**, *9*, 21.

(103) Husic, B. E.; Pande, V. S. *J. Chem. Theory Comput.* **2017**, *13*, 963–967.

(104) Dubes, R. C. *Pattern Recognit.* **1987**, *20*, 645–663.

(105) Kelley, L. A.; Gardner, S. P.; Sutcliffe, M. J. *Protein Eng. Des. Sel.* **1996**, *9*, 1063–1065.

(106) Pettersen, E. F.; Goddard, T. D.; Huang, C. C.; Couch, G. S.; Greenblatt, D. M.; Meng, E. C.; Ferrin, T. E. *J. Comput. Chem.* **2004**, *25*, 1605–1612.

(107) Langfelder, P.; Zhang, B.; Horvath, S. *Bioinformatics* **2007**, *24*, 719–720.

(108) Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; Lefebvre, E. *J. Stat. Mech.* **2008**, *2008*, P10008.

(109) Kabsch, W. *Acta Cryst. Sect. A* **1976**, *32*, 922–923.

(110) Harris, C. R. et al. *Nature* **2020**, *585*, 357–362.

(111) McKinney, W. In *Proceedings of the 9th Python in Science Conference*, ed. by van der Walt, S.; Millman, J., 2010, pp 56–61.

(112) Virtanen, P. et al. *Nat. Methods* **2020**, *17*, 261–272.

(113) Ward, J. H. *J. Am. Stat. Assoc.* **1963**, *58*, 236–244.

(114) R Core Team R: A Language and Environment for Statistical Computing; R Foundation for Statistical Computing, Vienna, Austria, 2017.

(115) Snedecor, G. W.; Cochran, W. G. *Iowa state University press, Ames, Iowa* **1989**.

(116) Wilcoxon, F. In *Springer Series in Statistics*; Springer New York: 1992, pp 196–202.

(117) Dunn, O. J. *J. Am. Stat. Assoc.* **1961**, *56*, 52–64.

(118) Watts, K. S.; Dalal, P.; Murphy, R. B.; Sherman, W.; Friesner, R. A.; Shelley, J. C. *J. Chem. Inf. Model.* **2010**, *50*, 534–546.

(119) ConfGen, Schrödinger LLC. Accessed 2021-02-03, 2020.

(120) Hjorth Larsen, A. et al. *J. Phys.: Condens. Matter* **2017**, *29*, 273002.

(121) Jacomy, M.; Venturini, T.; Heymann, S.; Bastian, M. *PLoS One* **2014**, *9*, e98679.

(122) Protein Preparation Wizard; Epik; Impact; Prime, Schrödinger LLC. Accessed 2021-02-03, 2020.

(123) Madhavi Sastry, G.; Adzhigirey, M.; Day, T.; Annabhimoju, R.; Sherman, W. *J. Comput. Aided. Mol. Des.* **2013**, *27*, 221–234.

(124) Frisch, M. et al. Gaussian˜16 Revision C.01, Gaussian Inc. Wallingford CT, 2016.

(125) Ebejer, J.-P.; Morris, G. M.; Deane, C. M. *J. Chem. Inf. Model.* **2012**, *52*, 1146–1158.

(126) Tosco, P.; Stiefl, N.; Landrum, G. *J. Cheminform.* **2014**, *6*, 37.

(127) Dewar, M. J.; Zoebisch, E. G.; Healy, E. F.; Stewart, J. J. *J. Am. Chem. Soc.* **1985**, *107*, 3902–3909.

(128) Stewart, J. J. *J. Comput. Aided Mol. Des.* **1990**, *4*, 1–103.

(129) Smith, D. G. et al. *J. Chem. Phys.* **2020**, *152*, 184108.

(130) Wishart, D. S. et al. *Nucleic Acids Res.* **2017**, *46*, D608–D617.

(131) Fessel, J. P.; Oldham, W. M. Nicotine Adenine dDinucleotides: The Redox Currency of the Cell, 2018.

(132) Cukier, R. I. *J. Phys. Chem. B* **2015**, *119*, 3621–3634.

(133) Cumeras, R.; Figueras, E.; Davis, C. E.; Baumbach, J. I.; Gracia, I. *Analyst* **2015**, *140*, 1376–1390.

(134) Da Silva, R. R.; Dorrestein, P. C.; Quinn, R. A. *Proc. Natl. Acad. Sci. USA* **2015**, *112*, 12549–12550.

(135) Gao, X.; Ramezanghorbani, F.; Isayev, O.; Smith, J. S.; Roitberg, A. E. *Journal of chemical information and modeling* **2020**, *60*, 3408–3415.

(136) Das, S.; Tanemura, K. A.; Dinpazhoh, L.; Keng, M.; Schumm, C.; Leahy, L.; Asef, C. K.; Rainey, M.; Edison, A. S.; Fernández, F. M., et al. *Journal of the American Society for Mass Spectrometry* **2022**, *33*, 750–759.

(137) Mölder, F.; Jablonski, K. P.; Letcher, B.; Hall, M. B.; Tomkins-Tinch, C. H.; Sochat, V.; Forster, J.; Lee, S.; Twardziok, S. O.; Kanitz, A., et al. *F1000Research* **2021**, *10*.

(138) Ropp, P. J.; Kaminsky, J. C.; Yablonski, S.; Durrant, J. D. *Journal of cheminformatics* **2019**, *11*, 1–8.

(139) Manathunga, Madushanka and Shajan, Akhil and Giese, Timothy J.; Cruzeiro, Vinícius Wilian D. and Smith, Jamie and Miao, Yipu and He, Xiping and Ayers, K and Brothers, E. and Götz, Andreas W. amd Merz, Kenneth Malcom QUICK, version 22.03, University of California San Diego, CA and Michigan State University, East Lansing, MI, 2022.

(140) Manathunga, M.; Miao, Y.; Mu, D.; Götz, A. W.; Merz Jr, K. M. *Journal of Chemical Theory and Computation* **2020**, *16*, 4315–4326.

(141) Miao, Y.; Merz Jr, K. M. *Journal of chemical theory and computation* **2015**, *11*, 1449–1462.

(142) Tanemura, K. A.; Das, S.; Merz Jr, K. M. *Journal of Chemical Information and Modeling* **2021**, *61*, 1647–1656.

(143) Ryu, S.; Kwon, Y.; Kim, W. Y. *Chemical Science* **2019**, *10*, 8438–8446.

(144) Gal, Y.; Ghahramani, Z. In *international conference on machine learning*, 2016, pp 1050–1059.

(145) PNNL Collision Cross Section Database.

(146) Picache, J. A.; Rose, B. S.; Balinski, A.; Leaptrot, K. L.; Sherrod, S. D.; May, J. C.; McLean, J. A. *Chemical science* **2019**, *10*, 983–993.

(147) Gal, Y.; Hron, J.; Kendall, A. *Advances in neural information processing systems* **2017**, *30*.

(148) He, K.; Zhang, X.; Ren, S.; Sun, J. In *Proceedings of the IEEE international conference on computer vision*, 2015, pp 1026–1034.

(149) Zhao, R.; Wang, M.; Chen, J.; Tong, Y.; Wei, G.-W. *Bulletin of mathematical biology* **2020**, *82*, 1–38.

APPENDIX

Table A1: Residue type assigned to each residue. Interaction type was assigned based on the type of residues interacting. '1' signifies the residue belongs to the category, while '0' signifies the residue is absent from the category. Flexible was defined as having a side chain with three consecutive free rotating bonds between heavy atoms. Small was defined as having a side chain of one carbon or less.

residue	anionic	cationic	polar	nonpolar	aromatic	flexible	small
Asp	1	0	0	0	0	0	0
Glu	1	0	0	0	0	1	0
Arg	0	1	0	0	0	1	0
Lys	0	1	0	0	0	1	0
His	0	0	1	0	1	0	0
Ser	0	0	1	0	0	0	1
Thr	0	0	1	0	0	0	0
Asn	0	0	1	0	0	0	0
Gln	0	0	1	0	0	1	0
Cys	0	0	1	0	0	0	1
Gly	0	0	0	1	0	0	1
Pro	0	0	0	1	0	0	0
Ala	0	0	0	1	0	0	1
Val	0	0	0	1	0	0	0
Ile	0	0	0	1	0	1	0
Leu	0	0	0	1	0	1	0
Met	0	0	0	1	0	1	0
Phe	0	0	0	1	1	0	0
Tyr	0	0	0	1	1	0	0
Trp	0	0	0	1	1	0	0

Table A2: Metabolites selected from the Human Metabolome Database for benchmarking the performance of AutoGraph. The name deposited in the HMDB was used without modification.

	name
HMDB0031506	4,4'-Diaminodibutylamine
HMDB0036938	1-O-Feruloylglucose
HMDB0033217	(2xi,6xi)-7-Methyl-3-methylene-1,2,6,7-octanetetrol
HMDB0037340	Cosmosiin
HMDB0032957	2-O-Feruloyltartronic acid
HMDB0032942	Momorcharaside A

Table A2 (cont'd)

HMDB0032921	Ethyl 2-furanpropionate
HMDB0037404	6-Glucopyranosylprocyanidin B1
HMDB0032752	R-2-Propenyl 1-propenesulfinothioate
HMDB0040659	Hoduloside VII
HMDB0031896	Erinacine D
HMDB0040706	7-Epi-12-hydroxyjasmonic acid glucoside
HMDB0031639	Glycerol 1-propanoate
HMDB0038265	3,3',5-Trihydroxy-4'-methoxy-6,7-methylenedioxyflavone 3-glucuronide
HMDB0031422	cis-Piceid
HMDB0031026	Calenduloside H
HMDB0030760	Pteroside B
HMDB0030693	Delphinidin 3,5-diglucoside
HMDB0030581	Perillanin
HMDB0040831	Rheinoside C
HMDB0030472	4-Ipomeanol
HMDB0030422	xi-Linalool 3-[rhamnosyl-(1-¿6)-glucoside]
HMDB0030406	Homomethionine
HMDB0037426	Astragalin 7-rhamnoside
HMDB0040960	Capsicoside B2
HMDB0040135	Hydroxydestruxin B
HMDB0039746	4"-O-Acetylafzelin
HMDB0029773	6S,9R-Dihydroxy-4,7E-megastigmadien-3-one 9-[apiosyl-(1-¿6)-glucoside]
HMDB0033391	C.I. Acid Green 5
HMDB0035475	Simmondsin 2'-ferulate
HMDB0039178	1,2-Digalloyl-beta-D-glucopyranose
HMDB0039110	(2S,3'S)-alpha-Amino-2-carboxy-5-oxo-1-pyrrolidinebutanoic acid
HMDB0034843	Acteoside
HMDB0039102	N-Carboxyacetyl-D-phenylalanine
HMDB0036339	25-Acetyl-6,7-didehydrofevicordin F 3-[glucosyl-(1-¿6)-glucoside]
HMDB0034361	Olitorin
HMDB0000094	Citric acid

Table A2 (cont'd)

HMDB0034081	Brassinolide
HMDB0034072	Gentianose
HMDB0033991	Daidzin
HMDB0036476	Araliasaponin I
HMDB0033865	Lotaustralin
HMDB0033839	Eudesmic acid
HMDB0039069	(R)-Byakangelicin 3'-glucoside
HMDB0033793	4-Heptenoic acid
HMDB0033750	D-Glycero-D-galacto-heptitol
HMDB0033741	Puddumin A
HMDB0036634	Phlorizin
HMDB0036933	1-Methyl 2-galloylgalactarate
HMDB0033508	Gonyautoxin IV
HMDB0039669	2-Hexyl-1,3-dioxolane-4-methanol
HMDB0033392	C.I. Acid Red 13
HMDB0030160	Natsudaidain
HMDB0041129	6"-O-(3-Hydroxy-3-methylglutaroyl)astragalin
HMDB0035745	Jujuboside B
HMDB0060833	N-Acetylserotonin glucuronide
HMDB0000210	Pantothenic acid
HMDB0013680	Caftaric acid
HMDB0000230	N-Acetylneuraminic acid
HMDB0060642	N-depropylpropafenone
HMDB0003339	D-Glutamic acid
HMDB0003265	Hesperidin
HMDB0003249	Rutin
HMDB0060797	6alpha,9alpha-Difluoroprednisolone-17-butyrate
HMDB0060813	Descarbonyl-lacosamide
HMDB0001846	Tetrahydrofolic acid
HMDB0038426	Butyl glucosinolate
HMDB0037892	Nomilinic acid 17-glucoside

Table A2 (cont'd)

HMDB0001451	(R)-lipoic acid
HMDB0001438	25-Hydroxyvitamin D2
HMDB0061054	Desacetylvinblastine
HMDB0060883	7-Hydroxy-R-acenocoumarol
HMDB0000625	Gluconic acid
HMDB0060890	Hydroxyterbinafine
HMDB0038502	Bisacurone B
HMDB0060921	3-Hydroxyibuprofen
HMDB0000201	L-Acetylcarnitine
HMDB0061108	3'-Hydroxybuspirone
HMDB0014720	Voriconazole
HMDB0014813	Tamoxifen
HMDB0037429	Astragalin
HMDB0041300	PRE
HMDB0041343	Chinenoside III
HMDB0029311	Deltonin
HMDB0041360	Prenyl arabinosyl-(1-¿6)-glucoside
HMDB0037478	4beta-(2-Aminoethylthio)epicatechin 3-gallate
HMDB0015548	Valganciclovir
HMDB0015542	Pivampicillin
HMDB0015303	Kanamycin
HMDB0015286	Gemifloxacin
HMDB0041515	Benzyl gentiobioside
HMDB0015132	Doxorubicin
HMDB0015054	Almotriptan
HMDB0000099	L-Cystathionine
HMDB0014950	Phenylbutazone
HMDB0014934	Candesartan
HMDB0000182	L-Lysine
HMDB0041665	3-Hydroxy-4-methoxyphenyllactic acid
HMDB0041707	Caffeic acid 4-O-glucuronide

Table A2 (cont'd)

HMDB0000448	Adipic acid
HMDB0039111	L-Acetopine
HMDB0038541	Goshonoside F5
HMDB0038751	Avenic acid B
HMDB0038992	Cistocardin
HMDB0038630	Niazimin
HMDB0038672	Lyciumin A
HMDB0038927	Isolariciresinol 9'-O-beta-D-glucoside
HMDB0038708	Citrusin C
HMDB0038562	8-Propanoylneosolaniol
HMDB0000067	Cholesterol
HMDB0039179	1,6-Digalloyl-beta-D-glucopyranose
HMDB0041159	Cyanidin 3-O-[b-D-Xylopyranosyl-(1-¿2)-[b-D-glucopyranosyl-(1-¿6)]-b-D-galactopyranoside]
HMDB0060935	4-trans-Hydroxyglipizide
HMDB0060910	8-Hydroxy-delta-9-THC
HMDB0060870	RPR112698
HMDB0060838	N-Desmethyl-p-hydroxyrosiglitazone
HMDB0060621	n-Demethylated piperazine
HMDB0060583	Trandolaprilat
HMDB0041728	(-)-Epicatechin 3'-O-glucuronide
HMDB0041552	2-[4-(3-Hydroxypropyl)-2-methoxyphenoxy]-1,3-propanediol 1-xyloside
HMDB0041513	Isopropyl apiosylglucoside
HMDB0041211	Sesaminol glucosyl-(1-¿2)-[glucosyl-(1-¿6)]-glucoside
HMDB0040937	Lactitol
HMDB0039293	Medicoside J
HMDB0040623	3'-(6"-Galloylglucosyl)-phloroacetophenone
HMDB0040575	Butyl formate
HMDB0040418	Majonoside R2
HMDB0040181	Fagopyritol A2
HMDB0040122	Madecassoside

Table A2 (cont'd)

HMDB0039640	Ustiloxin A
HMDB0039509	trans-p-Coumaric acid 4-glucoside
HMDB0039488	Chrysophanol 8-gentiobioside
HMDB0039458	Di-2-propenyl heptasulfide
HMDB0039332	2"-(6"-p-Coumaroylglucosyl)quercitrin
HMDB0038301	Apigenin 7-[feruloyl-(-¿2)-glucuronyl-(1-¿2)-glucuronide] 4'-glucuronide
HMDB0034249	Caryoptosidic acid
HMDB0038143	Calamin
HMDB0038042	Isovitexin 2"-O-(6"'-feruloyl)glucoside
HMDB0030388	9-(beta-D-Ribofuranosyl)zeatin
HMDB0030246	Narceine
HMDB0030104	Humulinic acid A
HMDB0029548	Diosmin
HMDB0029517	Limocitrin 3-glucoside
HMDB0029496	Fukiic acid
HMDB0029408	Betanin
HMDB0029258	Luteolin 6-C-glucoside 8-C-arabinoside
HMDB0015570	Fusidic Acid
HMDB0015169	Procainamide
HMDB0015002	Tacrolimus
HMDB0014911	Etoposide
HMDB0014906	Olopatadine
HMDB0014412	Cefmenoxime
HMDB0014344	Erythromycin
HMDB0010358	17-beta-estradiol 3-sulfate-17-(beta-D-glucuronide)
HMDB0006270	Linoelaidic acid
HMDB0003164	Chlorogenic acid
HMDB0001876	Epinephrine sulfate
HMDB0001852	all-trans-Retinoic acid
HMDB0001542	N-Acetyllactosamine
HMDB0001262	Maltotriose

Table A2 (cont'd)

HMDB0001220	Prostaglandin E2
HMDB0000570	Coproporphyrin III
HMDB0000488	(4E,15Z)-Bilirubin
HMDB0000482	Caprylic acid
HMDB0000446	N-alpha-Acetyl-L-lysine
HMDB0000417	3D,7D,11D-Phytanic acid
HMDB0000193	Isocitric acid
HMDB0030542	Neohesperidin dihydrochalcone
HMDB0031876	Prenyl glucoside
HMDB0031985	gamma-Glutamyl-S-methylcysteine
HMDB0034945	Stevioside
HMDB0037977	Cyanidin 3-galactoside
HMDB0037960	Quercetin 3-(2G-glucosylrutinoside)
HMDB0037635	xi-1-Ethoxy-1-butoxyethane
HMDB0037537	Quercetin 3-glucosyl-(1-¿2)-galactoside
HMDB0037499	3'-N'-Acetylfusarochromanone
HMDB0037491	Liquiritin apioside
HMDB0037427	Kaempferol 3-sophoroside 7-rhamnoside
HMDB0037415	Isomargaritene
HMDB0037332	8-Hydroxyluteolin 4'-methyl ether 8-glucoside
HMDB0036484	Trigoneoside Xb
HMDB0035872	Oleuropein
HMDB0035857	Ginsenoside A2
HMDB0035778	Ginsenoside Rd
HMDB0034862	4',6'-Dihydroxy-2'-methoxyacetophenone 6'-glucoside
HMDB0032590	Zingerone
HMDB0034649	Soyasaponin I
HMDB0034613	Torachrysone 8-beta-gentiobioside
HMDB0033908	(±)-2-Heptanol
HMDB0033838	Diethyl succinate
HMDB0033597	4-(4-Methoxyphenyl)-2-butanone

Table A2 (cont'd)

HMDB0033578	Aquifoliunine EIII
HMDB0033463	Subaphylline
HMDB0033320	Yessotoxin
HMDB0033282	8-Hydroxypinoresinol 4-glucoside
HMDB0033226	(1S,4R)-10-Hydroxyfenchone glucoside
HMDB0033201	9-Hydroxydecanoic acid
HMDB0033075	Methyl 1-(methylsulfinyl)propyl disulfide
HMDB0032796	Methyl 3-(2,3-dihydroxy-3-methylbutyl)-4-hydroxybenzoate
HMDB0061142	Fenoprofen glucuronide

Benchmark Selection

CCS values were collected from various databases. Only entries with number of heavy atom within 30 were retained. The Morgan fingerprint was calculated for all entries with 2048 bits and connectivity of 3. For each adduct, the Tanimoto similarity was calculated comprehensively between all entries. The produced similarity matrix was then hierarchically clustered using the Ward method. A threshold was applied to produce 20 distinct clusters. From each cluster, a centroid compound was selected by selecting the compound with the greatest within-cluster weighted degree. The benchmark set was manually curated thereafter from the centroids, removing structures which lacked stereochemical information to produce the final benchmark set.

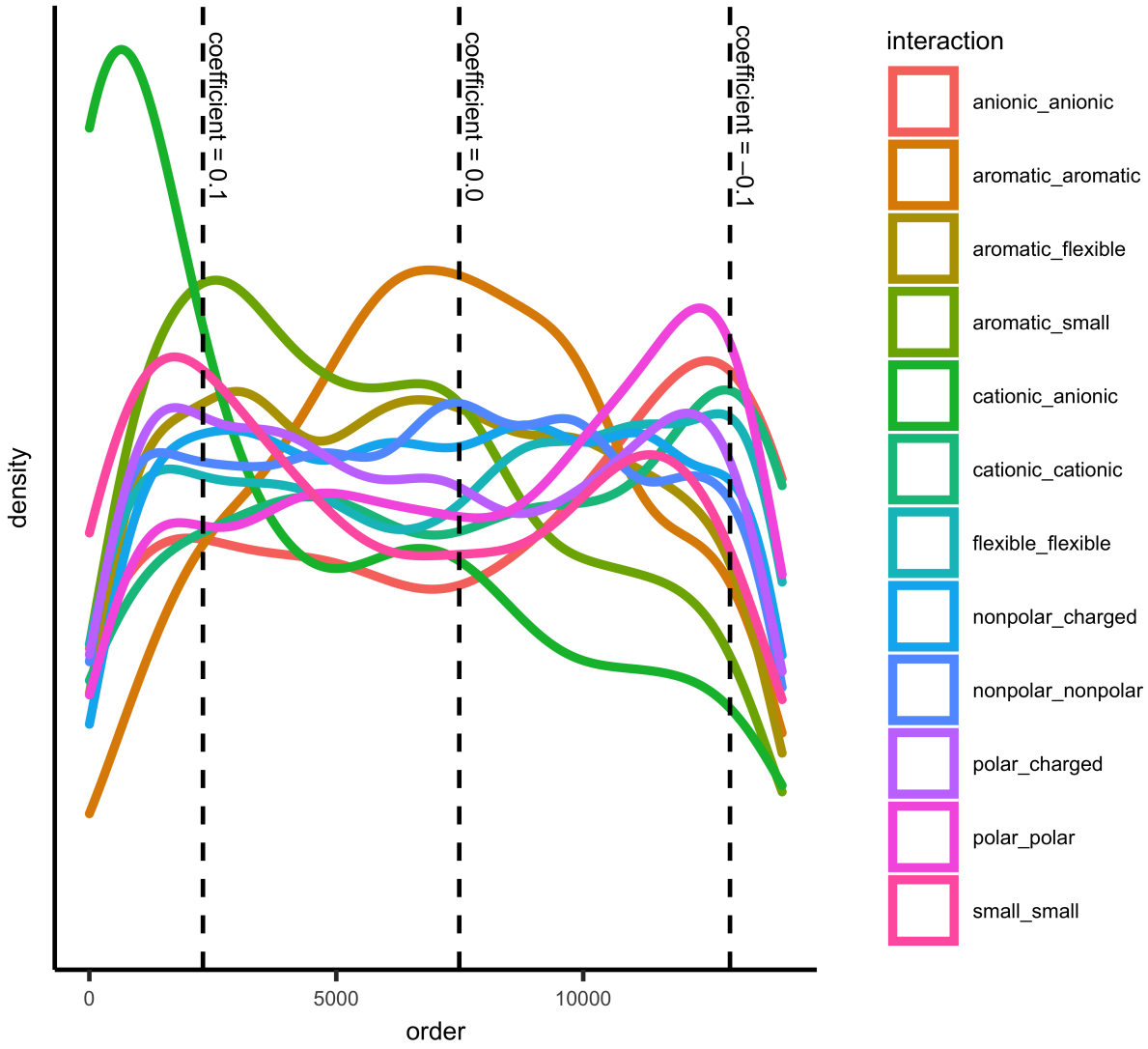

Figure A1: Density of various types of interactions applied to the coefficients ordered in decreasing order. The coordinates for coefficient values 0.1, 0.0, and −0.1 are specified.

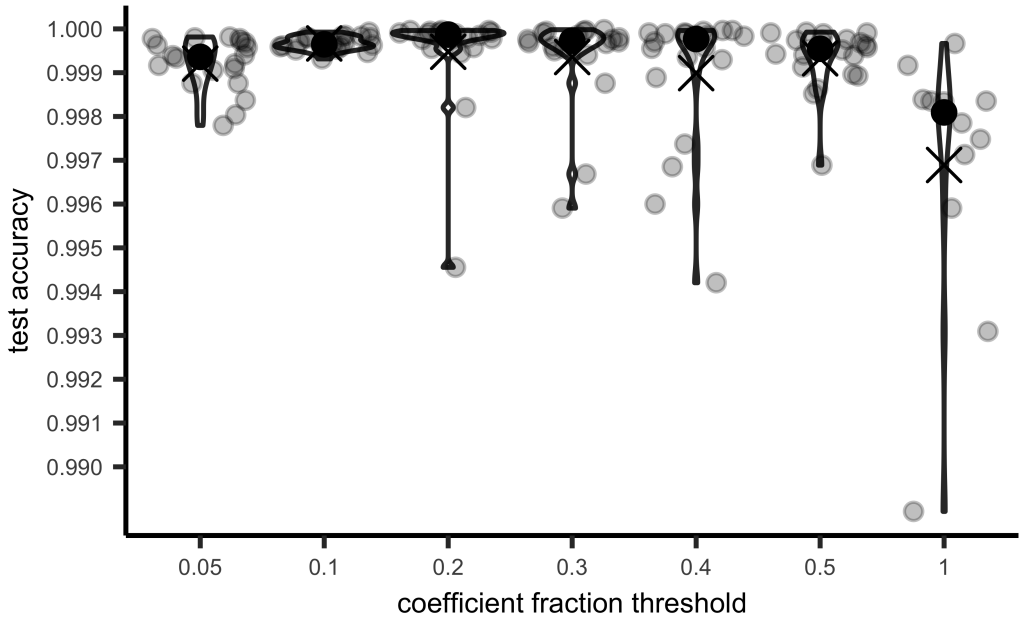

Figure A2: Distribution of test accuracies for LR classifiers trained on 0.9 of dataset, employing various thresholds of top features. Light gray dots are the raw data. Mean (×) and median (•) for each fraction are reported.

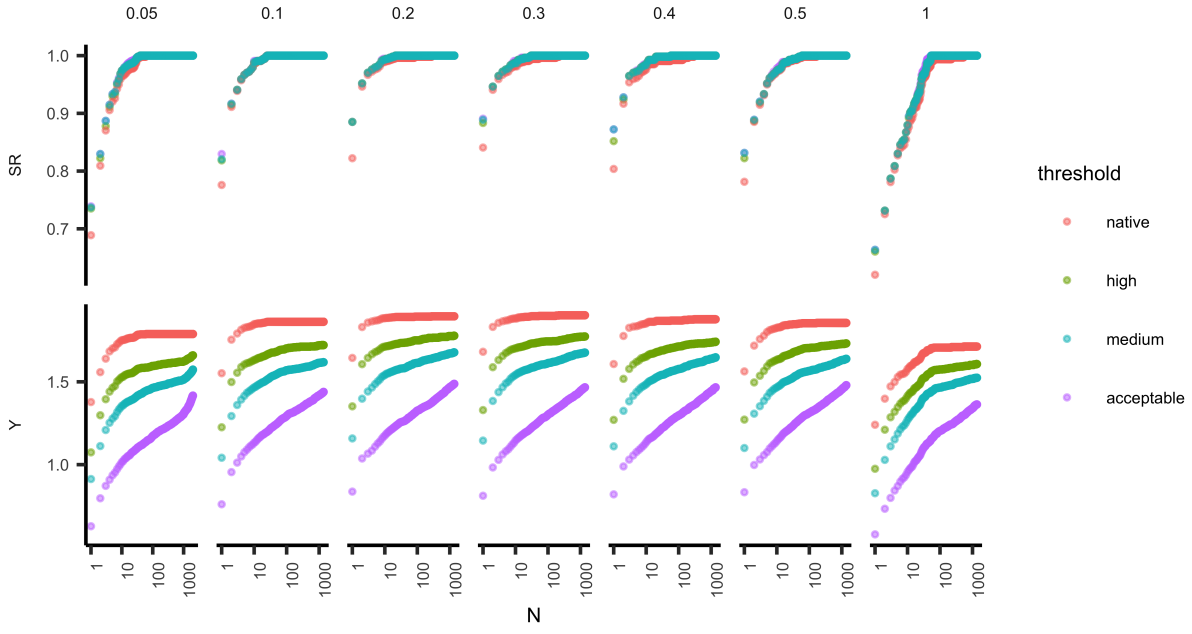

Figure A3: Comparison of success rate (top) and modified success rate (bottom) of the LR scoring function using various coefficient threshold and near-native structures.

Table A3: Representative structures were selected from available CCS databases.

name	CAS	charge	experimental CCS (Å2)
guanosine	118-00-3	1	165
Lobendazole	6306-71-4	1	141.7
L-tryptophan	73-22-3	1	150.3
ethanone	552-41-0	1	124.9
L-phenylalanine	63-91-2	1	139.8
norfloxacin	70458-96-7	1	184.2
acacetin	480-44-4	1	163.7
L-isoleucine	73-32-5	1	127.6
theophylline	58-55-9	1	143.1
sulfapyridine	144-83-2	1	151.6
R-proline	344-25-2	1	125.4
pyridoxine	65-23-6	1	134.7
adrenic acid	28874-58-0	1	189.4
4-methylpentanoic acid	646-07-1	–1	127.5
L-tryptophan	73-22-3	–1	158.6
salicylic acid	69-72-7	–1	120.7
3-methoxy-L-tyrosine	300-48-1	–1	150.9
1-methylxanthine	6136-37-4	–1	140.9
L-histidine	71-00-1	–1	128.5
R-proline	344-25-2	–1	122.5
biochanin A	491-80-5	–1	170.4
R-hesperetin	24604-97-5	–1	178.5

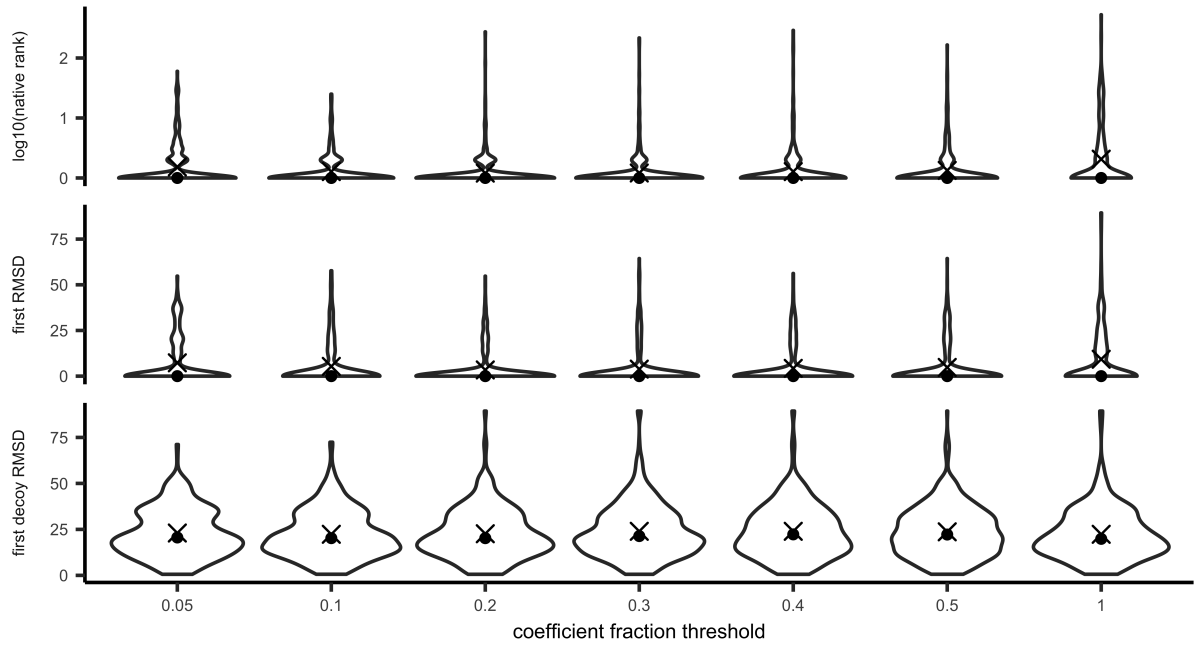

Figure A4: Comparison of performance of LR scoring function using various coefficient threshold via distributions of native rank (top), first RMSD (middle), and first decoy RMSD (bottom). Mean (×) and median (•) are reported for each distributions.

Table A4: Test set metabolites with independently measured CCS values in monoprotonated/monodeprotonated modes.

	CCS ($\AA^2$)	
name	[M+H]+	[M-H]-
folic acid	193.94	187.30
3-hydroxybutyric acid	-	129.91
oxypurinol	124.88	116.95
L-threonine	123.88	114.24
4-imidazoleacrylic acid	128.89	114.43
D-arginine	131.73	130.52
Fmoc-Val-OH	172.99	185.74
α-cyclodextrin	272.40	287.21
cyclo(Leu-Gly)	138.28	-
cis-urocanic acid	124.25	115.90
azithromycin	263.63	269.66
1,5-diaminonaphthalene	137.43	-
glutathione	167.94	157.80
3-indolebutyric acid	139.81	146.95
α-cyano-4-hydroxycinnamic acid	140.43	129.80
3-hydroxycoumarin	-	122.87
Fmoc-Pro-OH	168.75	-
artesunate	-	183.43

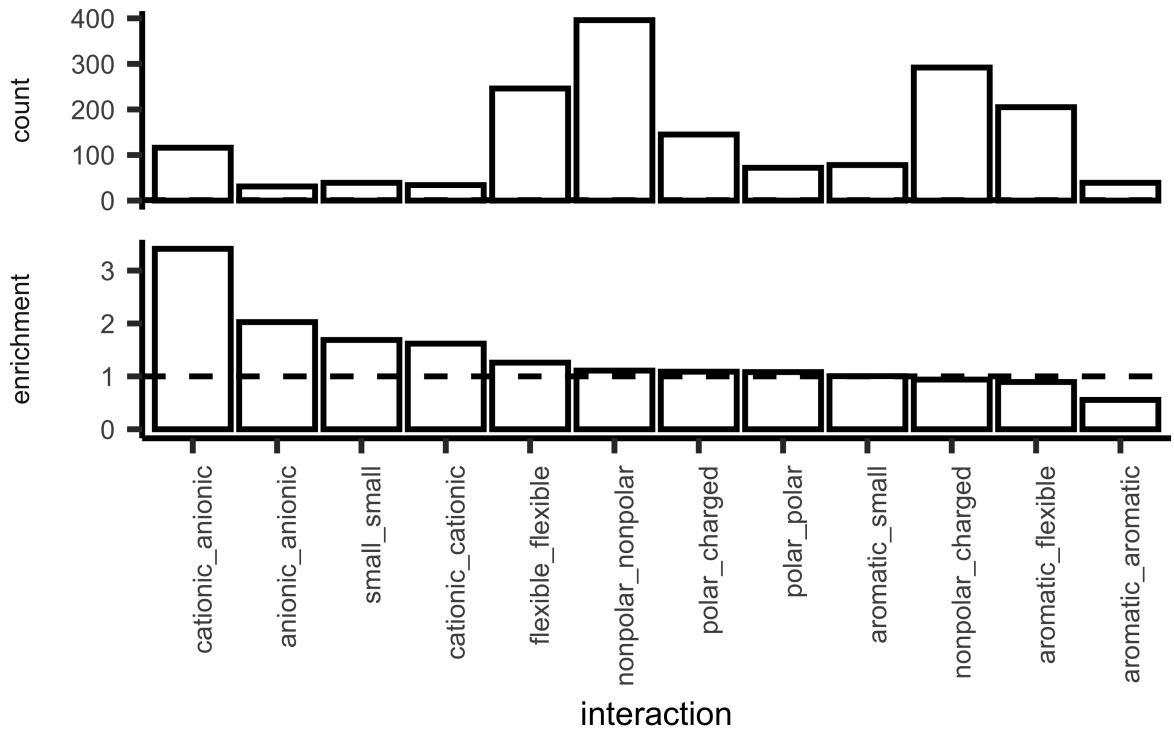

Figure A5: Composition of the top 0.1 salient features (1403 features) were summarized using the categories defined in Table A1. The appearance of interactions in each category was counted (top). However, number of features per category was not uniform. Fold enrichment of each category was summarized to account for the heterogeneity in number of features per category (bottom). The fold enrichment is the quotient of observed fraction and expected fraction, and describes the factor by which the category is overrepresented compared to if it were sampled by chance.

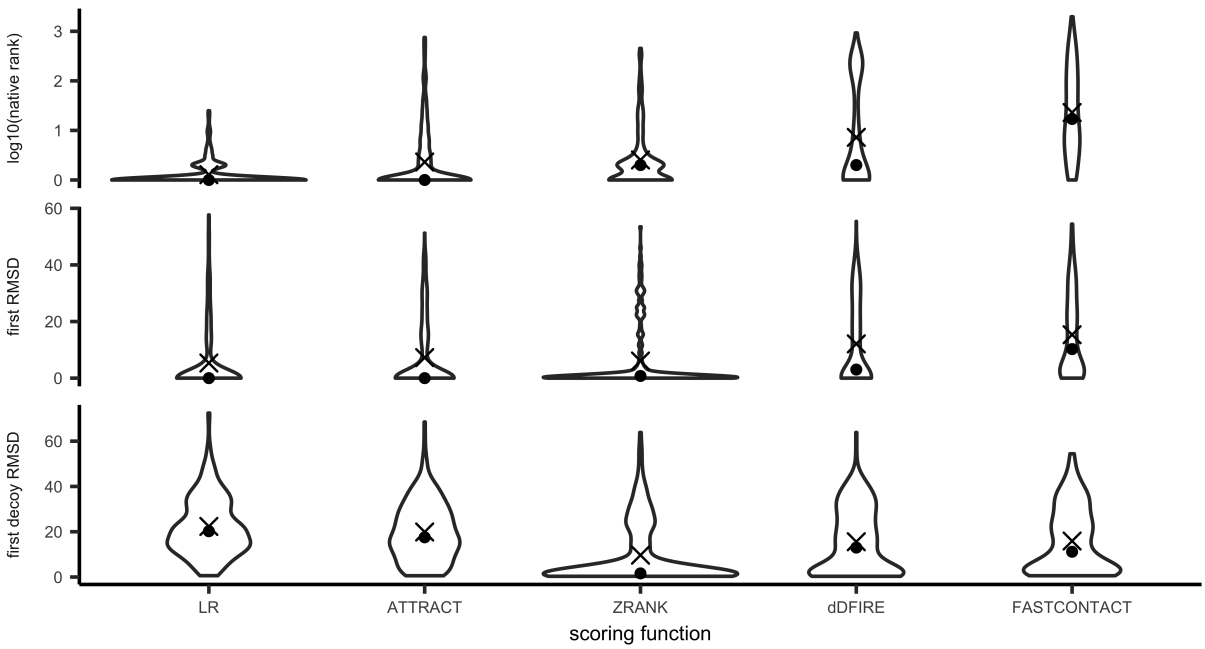

Figure A6: Comparison of scoring functions via by distributions of native rank (top), first RMSD (middle), and first decoy RMSD (bottom). Mean (×) and median (•) is reported for each distributions.

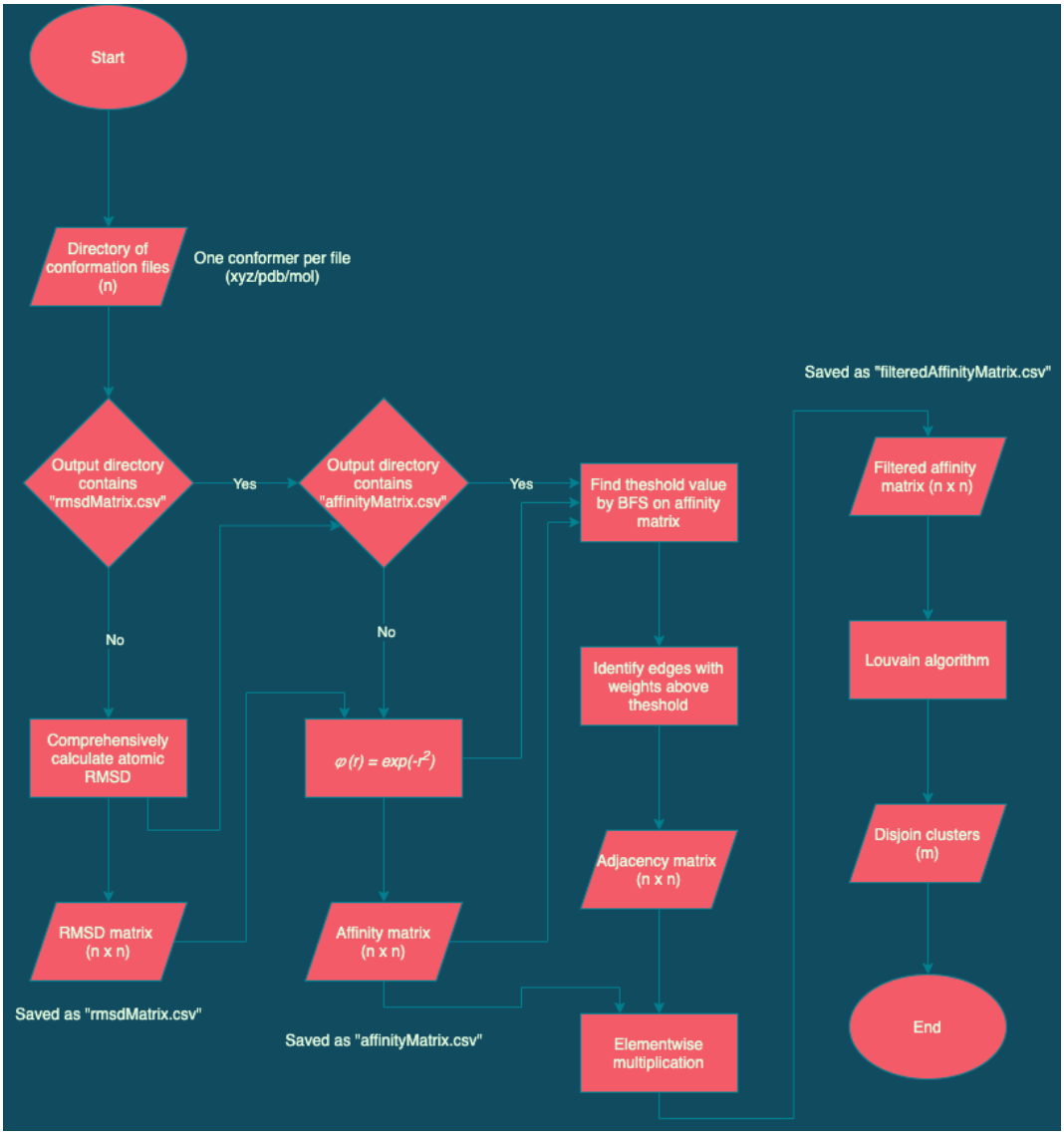

Figure A7: The AutoGraph protocol is summarized in a flowchart. In addition to the intermediate files shown, the output directory contains "cluster_summary.csv" and "communityStats.csv". "cluster_summary.csv" provides conformer-wise cluster assignment and centroid selection. "communityStats.csv" describes cluster-wise maximum and mean RMSD values.

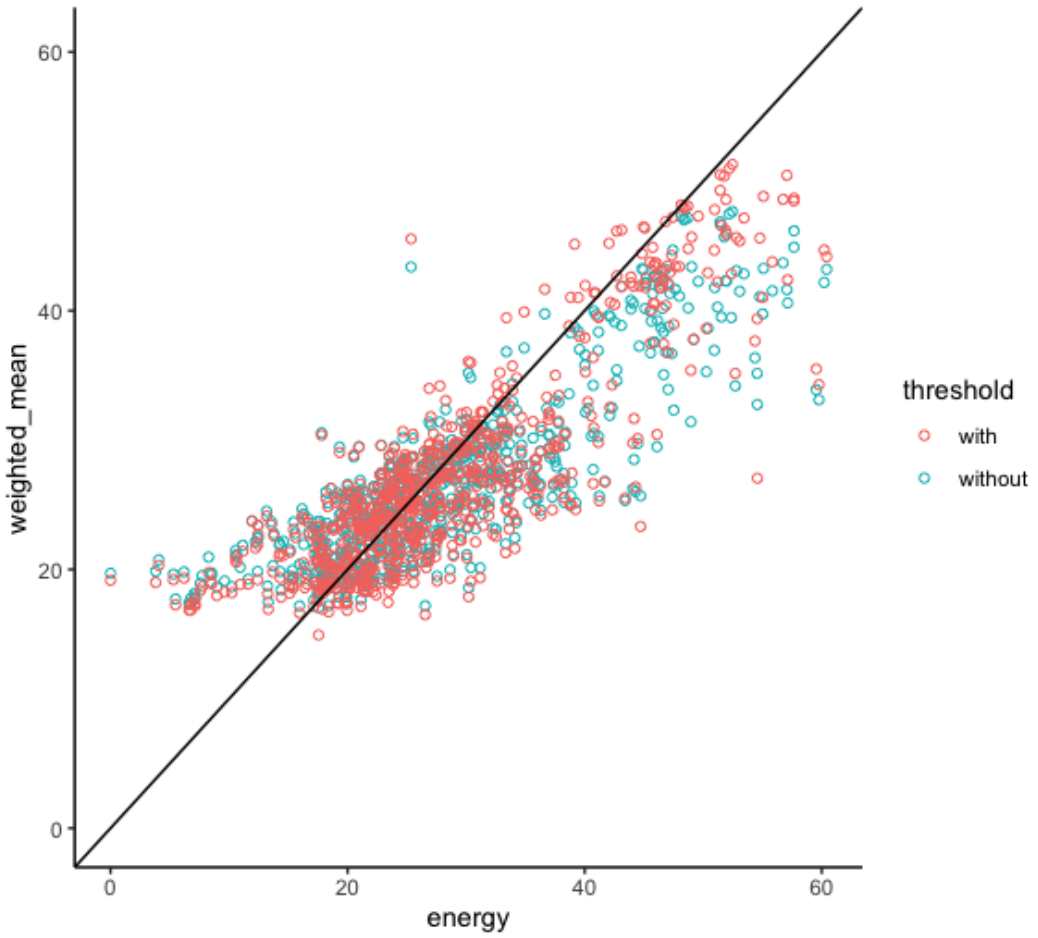

Figure A8: NAD+ exhibited a positive, monotonic relationship between actual and locally weighted estimated energy values with or without a weight threshold applied to the conformational graph($n = 785, \rho_{with} = 0.845, \rho_{without} = 0.842$). The line for x = y is plotted.

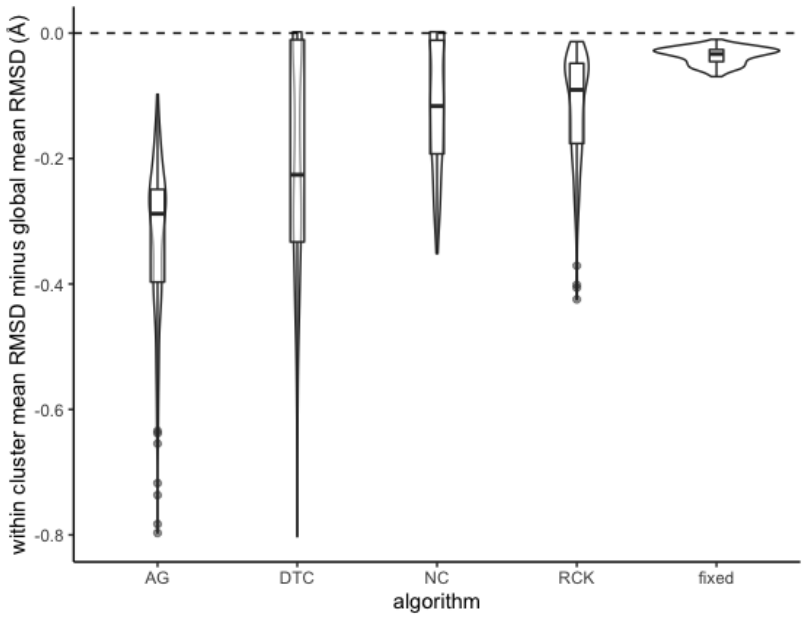

Figure A9: The distributions of within-cluster mean RMSD minus global mean RMSD is reported for each metabolite assessed by various conformational clustering algorithm (n = 200 per algorithm). Abbreviations: AutoGraph (AG), Dynamic Tree Cut/Ward hierarchical clustering (DTC), NMRCLUST (NC), Representative Conformer K-means (RCK), Ward hierarchical clustering with 18 clusters (fixed).

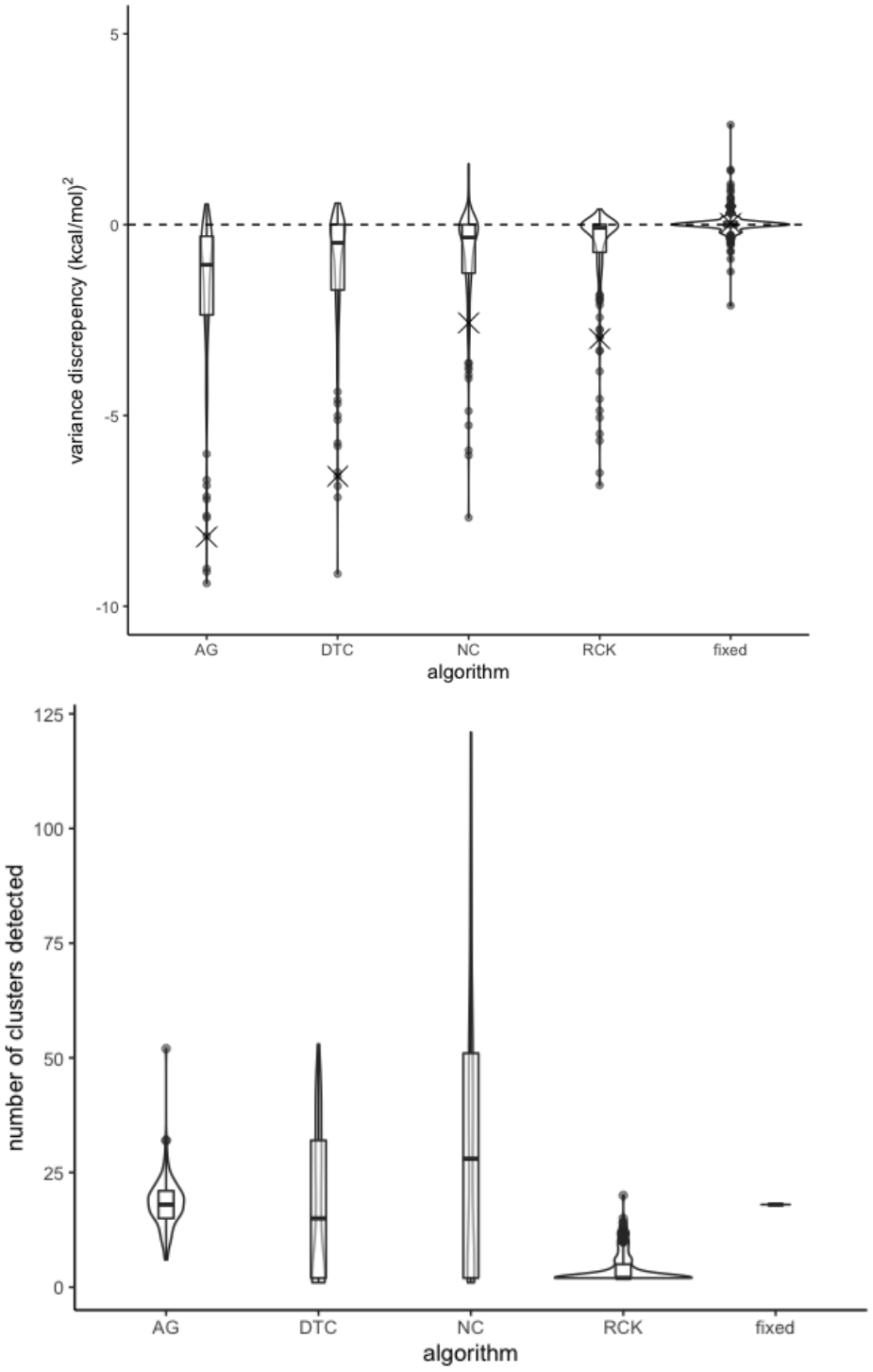

Figure A10: The distributions of discrepancy in variance of energy within clusters and overall ($\sigma^2_{within} - \sigma^2$) is reported for each metabolite assessed by various conformational clustering algorithm (top). The plot leaves out data points beyond values shown for clarity. The mean is indicated for each distribution ($\times$), which was calculated including the data points beyond the range visualized. Distribution of number of clusters detected is shown for each algorithm as well (bottom). Abbreviations: AutoGraph (AG), Dynamic Tree Cut/Ward hierarchical clustering (DTC), NMRCLUST (NC), Representative Conformer K-means (RCK), Ward hierarchical clustering with 18 clusters (fixed).

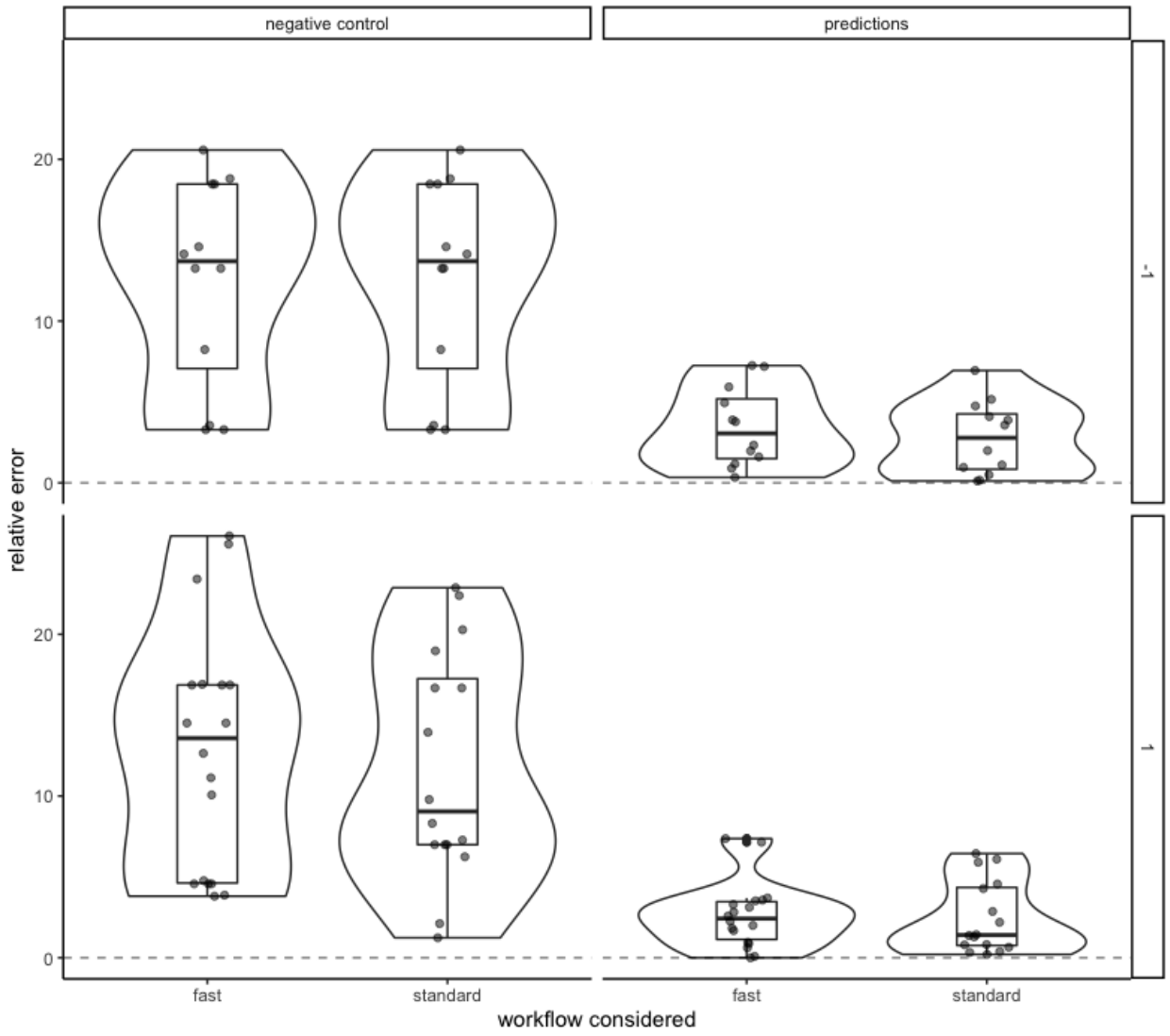

Figure A11: The fast and standard workflows were evaluated in terms of mean relative error against a negative control. The distributions were grouped by the charge of the ion considered. Each point corresponds to the relative error of exactly one structure included in the benchmark set.