Predicting residue-residue contacts at protein-protein interfaces using surface features - a machine learning approach

A Thesis

submitted to

Indian Institute of Science Education and Research Pune in partial fulfilment of the requirements for the BS-MS Dual Degree Programme

by

Adithyan Unni

IISER PUNE

Indian Institute of Science Education and Research Pune

Dr. Homi Bhabha Road,

Pashan, Pune 411008, INDIA.

May 2023

Supervisor: Dr. Carlos Óscar Sorzano Sánchez

Centro Nacional de Biotecnología (CSIC), Madrid

Certificate

This is to certify that this dissertation entitled ‘Predicting residue-residue contacts at protein-protein interfaces using surface features - a machine learning approach’ towards the partial fulfilment of the BS-MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents study/work carried out by Adithyan Unni at the Indian Institute of Science Education and Research, Pune, under the supervision of Dr. Carlos Óscar Sorzano Sánchez, Principal Investigator, Centro Nacional de Biotecnología (CSIC), Madrid, during the academic year 2022-2023.

SORZANO SANCHEZ CARLOS OSCAR - 25669302Z

Firmado digitalmente por SORZANO SANCHEZ CARLOS OSCAR - 25669302Z
Fecha: 2023.05.09 00:25:04 +02'00'

Dr. Carlos Óscar Sorzano Sánchez

Committee:

Dr. Carlos Óscar Sorzano Sánchez

Dr. M.S. Madhusudhan

This thesis is dedicated to my grandparents.

Declaration

I hereby declare that the matter embodied in the report entitled ‘Predicting residue-residue contacts at protein-protein interfaces using surface features - a machine learning approach’ are the results of the work carried out by me at the Indian Institute of Science Education and Research, Pune, under the supervision of Dr. Carlos Óscar Sorzano Sánchez, Principal Investigator, Centro Nacional de Biotecnología (CSIC), Madrid and the same has not been submitted elsewhere for any other degree.

Unni Adithyan

Adithyan Unni

Date: 6th May 2023

Table of Contents

List of Tables

List of Figures

Abstract

Proteins interact with other macromolecular targets, such as small molecules, nucleic acids, and other proteins, via their surfaces. Protein-protein interactions are likely to be influenced by the geometrical and physicochemical properties of the surfaces of the interacting proteins. To take advantage of this for the purpose of protein-protein interface prediction, we modify BIPSPI, an XGBoost-based partner-specific protein interface predictor, using geometrical and chemical features extracted from protein surfaces in the form of patches by means of MaSIF, a framework for the extraction of meaningful features from the surfaces of proteins. We construct a map from the surface-patch level representation constructed by MaSIF to the residue-pair representation used by BIPSPI. We show that the addition of internally sorted protein surface patches to BIPSPI's existing residue-pair representation increases the mean ROC-AUC performance of the existing predictor from 0.9153 to 0.9222 when evaluated with 10-fold cross-validation on a subset of Docking Benchmark v5.5. Additionally, we also evaluate the relative impact of the various features used in training on the performance of the combined model in terms of loss reduction over tree splits. We observe that sorting protein surface patches internally along the feature axes increases model performance and alters the relative impacts of various features. Furthermore, to reduce memory consumption while training with protein surface patches, we develop both principal component analysis-based and autoencoder-based approaches to patch compression. We observe that both methods exhibit competitive performance when trained with sorted patches but not unsorted patches.

Acknowledgements

At the outset, I would like to extend my gratitude to my supervisor, Dr. Carlos Óscar Sorzano Sánchez, for his extensive guidance and support throughout the course of my thesis. I am especially grateful for the opportunity he offered me to work on a problem I have been passionate about for years and for the freedom accorded to me in the time I have worked with him to pursue ideas I believed in. I would also like to express my sincere gratitude to Dr. M.S. Madhusudhan for his continued support over the time I have spent at IISER Pune. Whether it be a project on residue depth prediction, academic advice, or just to chat cool protein science, I consider myself fortunate to have had access to his office over the past few years. Together, Dr. Sorzano and Dr. Madhusudhan have set immeasurably high standards for the support and guidance I should expect from academic advisors in the future.

I would be remiss if I did not mention the wonderful folks I worked with in Spain. I would like to thank Dani, Mikel, Giedre, Fede, Borja, Oier, Irene, Patricia, Marcos, and everyone else at the B.13 for having fostered a lab environment that is incomparably supportive and hospitable. It is not often that one is excited to wake up at 5:00 AM to catch the train to work on a chilly December morning – Madrid felt like Mumbai in their warmth.

I am deeply indebted to my friends from IISER who have been by my side over the past five years, through the highs and lows, the hills and troughs. I'm grateful to Atreyi, Jatin, Chinmay, Shruthi, Suryadeepto, Siddharth, Shivani, Arjun, Chebi, Kunjal, Devjyoti, Vasudha, Aarcha, Rohit, and so many more for moulding me into the person I have become today.

None of this would have been possible without the love and support of my mum, dad, and grandparents. They have supported my dreams to pursue research from when I discovered chemistry videos on YouTube. I am eternally grateful for everything they have done for me. I continually strive to make them proud and hope to take them along as I journey through science.

Contributions

Contributor name	Contributor role
Adithyan Unni, Carlos Óscar Sorzano Sánchez	Conceptualization Ideas
Adithyan Unni	Methodology
Adithyan Unni	Software
Adithyan Unni	Validation
Adithyan Unni	Formal analysis
Adithyan Unni	Investigation
Carlos Óscar Sorzano Sánchez	Resources
Adithyan Unni	Data Curation
Adithyan Unni	Writing - original draft preparation
Adithyan Unni, Carlos Óscar Sorzano Sánchez	Writing - review and editing
Adithyan Unni	Visualization
Carlos Óscar Sorzano Sánchez	Supervision
Adithyan Unni, Carlos Óscar Sorzano Sánchez	Project administration
Carlos Óscar Sorzano Sánchez	Funding acquisition

This contributor syntax is based on the Journal of Cell Science CRediT Taxonomy[1].

[1] https://journals.biologists.com/jcs/pages/author-contributions

1 Introduction

Proteins play an integral role in many, if not all, biological processes critical to life. They are extensively involved in processes that occur at various levels of biological organization. In the form of enzymes, proteins mediate a wide range of chemical reactions critical for cellular metabolism (Jeong *et al.*, 2000). Cellular signalling pathways that allow cells to respond to both intracellular and extracellular stimuli are constituted of numerous proteinaceous components. Certain proteins undergo conformational changes upon molecular recognition events, such as binding small molecules or peptides, serving as switches for downstream processes (Milburn *et al.*, 1990; Ha and Loh, 2012; Ghusinga *et al.*, 2021; Alberstein *et al.*, 2022). By forming structurally robust polymers, proteins are also capable of providing mechanical support, both at the level of the cell in the form of elaborate cytoskeletal frameworks (Fuchs and Cleveland, 1998; Herrmann *et al.*, 2007; Fletcher and Mullins, 2010), and at the macroscopic level, as the building blocks of skeletal musculature (Huxley and Niedergerke, 1954; Clarke, 2010).

In many cases, the biological relevance of proteins arises from their ability to bind to a range of other biomolecules. Proteins associate with small molecules, carbohydrates, nucleic acids, and other proteins. Protein-protein interactions (PPIs) are a critical class of interactions owing to their prevalence in cellular signalling pathways and regulatory networks (Søgaard-Andersen and Valentin-Hansen, 1993; Pawson and Nash, 2000). Knowledge of a protein's binding partners, coupled with mutational studies, allows for researchers to determine the functional role it plays in the cell – this is of relevance i) at a fundamental level, in terms of elucidating the molecular processes that facilitate life and ii) from a clinical perspective, in working towards establishing mechanisms for diseases that either cause or arise from the dysregulation of these processes. Modern experimental techniques have allowed us to delve a step deeper by revealing the specific amino acid residues that proteins use to bind to each other. Mutations to a PPI's interface residues can either strengthen or weaken binding. Identifying the interface of a protein-protein interaction enables protein biochemists to perform mutational studies to determine which residues are most important for an interaction (Stites, 1997; Dall'Acqua *et al.*, 1998;

Zanotti *et al.*, 2008). Protein-protein interfaces are also of clinical interest owing to their functional significance and are often the targets of drugs designed to inhibit interactions (Scott *et al.*, 2016; Li *et al.*, 2017; Chen *et al.*, 2018). Interfaces have gained even more prominence in recent times in the context of novel methods to design custom protein-protein interactions (Kim *et al.*, 2021; Cao *et al.*, 2022; Marchand *et al.*, 2022). Improvements made in our ability to resolve and characterize protein-protein interfaces have the potential to greatly enhance the resolution at which we understand protein-mediated regulatory processes and advance the development of novel drugs and technologies to fight diseases.

1.1 Experimental methods for protein-protein interface prediction

The structures of protein-protein interfaces have traditionally been determined using experimental methods such as X-ray crystallography (Smyth and Martin, 2000; Bahadur and Zacharias, 2008) and Nuclear Magnetic Resonance (NMR) spectroscopy (Hu *et al.*, 2021). While such methods have been indispensable over the past few decades in advancing our understanding of how proteins interact, each method has its disadvantages. X-ray crystallography cannot be used to determine the structures of proteins (and, by extension, protein-protein complexes) that are difficult to crystallize (Zheng *et al.*, 2015; Harkey *et al.*, 2019). While desirable in its ability to resolve protein-protein interaction dynamics, NMR spectroscopy is limited in its capacity to determine the structures of large proteins at high resolution and is exceptionally motion-sensitive (Xue *et al.*, 2015). Over the past decade, cryo-electron microscopy (cryo-EM) has seen significant gains in popularity in light of its non-requirement of protein crystals and ability to capture multiple conformations in a single experiment (Bai *et al.*, 2015; Benjin and Ling, 2020). Even as resolution continues to improve, cryo-EM instrumentation remains prohibitively expensive for capital-sparse research environments with microscopes achieving high resolutions costing several millions, notwithstanding commensurate infrastructure and operating costs. All of the aforementioned techniques are labour-intensive and time-consuming and none are high-throughput. In light of these disadvantages, there is a space for fast, high-throughput, and accessible computational methods of protein interface prediction to exist.

1.2 Machine learning: A Brief Primer

Over the past decade, most computational methods of protein interface prediction have made extensive use of machine learning algorithms. ‘Machine learning’ (ML) methods are mathematical algorithms that make predictions on unseen data based on inferences made from data they are exposed to. Such methods have seen an enormous surge in popularity owing to generational improvements made in the fronts of parallel computing hardware and software and a monumental increase in the capacity of computing systems to collect and transfer data.

Machine learning and deep learning-based methods are particularly powerful owing to their capacity to derive meaning from complex forms of data. Deep learning methods are able to recognize patterns in datasets that are not apparent to humans and use these patterns to make predictions depending on what the prediction task is (LeCun *et al.*, 2015). They have proven exceptionally capable on a wide variety of prediction tasks involving myriad forms of data and have greatly influenced how we interact with and benefit from technology. Machine learning algorithms form the basis for voice recognition software, the recommender systems that power entertainment websites, and determine the advertisements we are shown on social media (Jordan and Mitchell, 2015; LeCun *et al.*, 2015; Sarker, 2021).

Prediction tasks that fall under the ambit of classification and regression are conventionally addressed with a class of machine learning algorithms called ‘supervised learning’ methods. In supervised learning, numerical representations of input data are provided to a learning algorithm, paired with corresponding labels (Tarca *et al.*, 2007; Greener *et al.*, 2022). Learning algorithms are mathematical models whose parameters are optimized over an iterative process called ‘training’. During a model’s training phase, its parameters are altered depending on how different its predicted outputs are from the true labels associated with the input data. These algorithms vary significantly in complexity and in the inductive biases associated with them. The term ‘inductive bias’ refers to the set of assumptions made by the algorithm to make predictions on unseen data (Goyal and Bengio, 2022). Choosing a specific learning algorithm to build a predictive model requires thorough domain knowledge on the programmer’s end to assess the required level of model complexity and the inductive biases most appropriate for the prediction task at hand.

Machine learning algorithms have evolved to accept input data of a wide variety of forms. Traditionally, each data point is represented mathematically as an array of numbers representing key features that may influence the target variable. For instance, a model designed to predict the value of a house might use square footage, number of bedrooms, and number of floors as input features. Models of greater complexity can accept more complex input data forms such as images, text, and audio clips. In some cases, feature engineering is an essential preliminary step that precedes training. Feature engineering is the process of pre-processing raw input data to extract features that are relevant to the prediction task at hand (Anderson and Cafarella, 2016).

The simple example of the linear model for regression illustrates most terms used in the above paragraph. Linear models for regression assume that the target variable is a linear combination of the input variables. The parameters for a linear model are the coefficients associated with the input variables. The optimal parameters for a linear model are usually those that minimize the sum of the squared errors computed between the true and predicted values for the points in the training dataset.

1.3 Machine learning for protein-protein interface prediction

Owing to their ability to make use of complex input data representations, machine learning methods lend themselves exceptionally well to prediction tasks in computational protein biology. Conventionally, there are two broad phrasings of the protein-protein interface prediction problem. These are the partner-independent and partner-specific approaches (Xue *et al.*, 2015). The two methods can be defined as follows:

Partner-independent interface prediction: Given a protein A, predict whether residue r_a belonging to protein A is part of the interface protein A forms with any other protein.

Partner-specific interface prediction: Given that proteins A and B bind, identify all pairs of residues (r_a, r_b), where r_a belongs to A and r_b belongs to B, such that r_a and r_b interact.

Partner-specific interface prediction allows for the identification of specific residue-pairs that are important for interactions to occur. Since they intrinsically require knowledge of both interacting partners, such predictors are more likely to yield reliable results in cases where a given protein has multiple binding partners (Xue *et al.*, 2015). While regarded as two separate categories, the core machine learning methodologies employed in both sets of models tend to be similar, with changes primarily arising from differences in training datasets and representation structure.

Machine learning methods in this space have traditionally used two broad classes of features: sequence-based features and structure-based features. Sequence-based features refer to properties of protein constituents (typically residues) that can be inferred directly or computed from their primary sequences. Examples of residue-level features include isoelectric point, amino acid identity, and residue conservation. Conservation has consistently proven to be a powerful feature for residue-residue contact prediction (Ovchinnikov *et al.*, 2014; Green *et al.*, 2021). The intuition underlying why conservation-based features are strong predictors of residue-residue contacts is that surface residues that are involved in the formation of complexes are more conserved than non-interface residues (Choi *et al.*, 2009; Teppa *et al.*, 2017). The extent of a residue's evolutionary conservation is usually quantified using its corresponding vector in the position-specific scoring matrix (PSSM) or hidden Markov model (HMM) profile corresponding to the protein's sequence (Eddy, 1995; Altschul, 1997; Finn *et al.*, 2011). These profiles are constructed through iterative multiple sequence alignments (MSAs) of the protein's sequence against a non-redundant sequence database such as UniRef or UniParc (Leinonen *et al.*, 2004; Suzek *et al.*, 2007). For a given residue, many predictors also include the PSSM vectors/HMM profiles of a window of sequentially neighbouring amino acids. Successful early sequence-only machine learning methods such as PPiPP (Ahmad and Mizuguchi, 2011) and PSIVER (Murakami and Mizuguchi, 2010) for protein-protein interface prediction solely utilized such representations in combination with simple machine learning architectures. While the performance exhibited by these predictors is modest in comparison to that of modern predictors, they illustrate the effectiveness of evolutionary features even when they are used with simple model architectures.

Pairwise protein sequence co-variation has also been used to predict residue-residue contacts at protein interfaces (Hopf *et al.*, 2014; Ovchinnikov *et al.*, 2014; Green *et al.*, 2021). Statistical models have been employed to analyze multiple sequence alignments of pairs of sequences for the presence of co-evolving residues (Seemayer *et al.*, 2014; Hopf *et al.*, 2019). The strength of co-evolution has been used in conjunction with monomer accessible surface area in logistic regression models to predict residue-residue contacts – Green *et al*. (2021) report a recall of 20.7% at a false positive rate of 0.1% on a custom held-out test set. Co-evolution information has also been used to construct proteome-level interaction networks for *E. coli* (Cong *et al.*, 2019).

Multiple sequence alignments in combination with attention-based neural network architectures (Vaswani *et al.*, 2017; Veličković *et al.*, 2018) have been used to great success on the task of protein structure prediction in the forms of AlphaFold2 (Jumper *et al.*, 2021) and RoseTTAFold (Baek *et al.*, 2021). AlphaFold2 has been recently adapted to make multimer-level predictions in the absence of structural information for both homomeric and heteromeric complexes through the use of multi-chain MSAs. Dubbed AlphaFold-multimer (Evans *et al.*, 2022), the model competently predicts the structures of a significant fraction of complexes in the Protein Data Bank (wwPDB consortium, 2019).

Conservation-based features are expected to falter in cases where the interacting proteins exhibit low homology to other known protein sequences. The computation of multiple sequence alignments is a resource-intensive step during testing and is often responsible for increased time taken during inference. Recent advancements made in the field of natural language processing have been extended to protein sequence datasets to account for these limitations. Unsupervised learning methods have been extensively leveraged to create numerical representations (called ‘embeddings’) of residues that are i) context-aware and ii) encapsulate their core properties (Rives *et al.*, 2021; Elnaggar *et al.*, 2022). To construct these representations, protein sequences extracted from large sequence databases are ‘corrupted’ – amino acids at random positions are replaced with ‘masks’. A learning method is then trained to predict the correct amino acid at a given position to high accuracy, using as input the other amino acids that constitute the sequence. The fully-trained model, sans the final prediction layer, is subsequently used with new input protein sequences to generate context-aware residue representations that can be used for various prediction tasks. ISPRED-SEQ (Manfredi *et al.*, 2023), a partner-independent sequence-only interaction site predictor, uses residue representations constructed using the ESM-1b (Rives *et al.*, 2021) and ProtT5 (Elnaggar *et al.*, 2022) protein language models in combination with a 1D Convolutional Neural Network and deep layers to achieve an ROC-AUC of 0.82 and MCC of 0.34 on a benchmark dataset of 448 protein chains. A similar model, EDLMPPI (Hou *et al.*, 2023), combines ProtT5-derived residue representations with PSSMs and physicochemical features and uses a bi-directional LSTM network for protein binding site prediction to achieve a similar ROC-AUC. It is to be noted that both methods are partner-independent.

The most competent methods of protein-protein interface prediction use structural information from the input proteins in addition to features derived at the sequence-level. A vast amount of geometrical information can be extracted from the structures of the interacting proteins and represented in highly-informative numerical representations. Structural descriptors commonly used to describe residues in interacting proteins include solvent accessible surface area, depth index, and protrusion index (Mihel *et al.*, 2008; Minhas *et al.*, 2014). These features can be computed directly from the coordinates and atom identifiers contained in a protein's PDB/mmCIF representation. The secondary structure a residue is involved in is also a commonly used feature in protein interface prediction and is assigned to residues using programs such as DSSP (Kabsch and Sander, 1983). Many partner-specific predictors use a residue-pair data representation, where the model is trained with residue-pair vectors that contain the features representing cross-protein pairs of residues. PAIRPred (Minhas *et al.*, 2014), one of the earliest partner-specific machine learning-based protein interface predictors, used a combination of conservation-based and structure-based features with a kernel SVM-based architecture to achieve a leave-one-complex-out performance of ROC-AUC 0.88 on Docking Benchmark v3.0 (Chen *et al.*, 2003; Hwang *et al.*, 2008). The use of graph convolutional neural networks (Kipf and Welling, 2017) for protein interface prediction was explored by Fout *et al*. (2017), where the input pair of proteins are interpreted as graphs and processed with graph convolutions. In this representation, residues constitute the nodes of the graph and are represented with structure-based and sequence-based features (Fout *et al.*, 2017). Such networks have also been modified to incorporate attention-layers and trained for the purpose of paratope and epitope prediction (Pittala and Bailey-Kellogg, 2020). Gradient-boosting-based methods have also seen considerable success in the field. BIPSPI (Sanchez-Garcia *et al.*, 2019, 2022) is an XGBoost-based (Chen and Guestrin, 2016) predictor that uses an extensive array of sequence-based and structure-based features to achieve an impressive ROC-AUC performance of 0.9057 on Docking Benchmark v5 (Vreven *et al.*, 2015). In addition to encoding the structure-based features of a residue-pair, the BIPSPI residue-pair vector representation also contains a summary of the feature values of the immediate neighbours of the focal residues defined by the Voronoi diagrams of the two proteins.

Significant advancements made in deep learning architectures and computing power have recently allowed for the development of protein interface predictors that extensively utilize the geometric information contained in the interacting protein partners. These predictors differentiate themselves from those that use structural information in a residue-pair context by using low-level structural information to create complex latent representations of their inputs that surpass hand-crafted features. PINet (Dai and Bailey-Kellogg, 2021), for instance, visualizes binary proteins as pairs of point clouds and uses an architecture inspired by PointNet (Qi *et al.*, 2017) to directly gleam geometrical information from the point clouds to avoid the disruption of geometry that may arise through processes such as 3D voxelization. Each point is represented by its coordinates in 3D space, its charge computed through Poisson-Boltzmann electrostatics, and hydrophobicity. The method uses a Spatial Transformer Network (Jaderberg *et al.*, 2015) trained on labelled pairs of point clouds to perform point-level prediction that can be mapped back to the residue-level. PeSTo (Krapp *et al.*, 2022) uses a similar point cloud geometric transformer-based approach for partner-independent protein interface prediction and achieves highly competitive performance on benchmark datasets, even in the absence of physicochemical features. Another popular method, MaSIF (Gainza *et al.*, 2020), operates at the level of the protein's surface, following the intuition that the shape and charge complementarity of protein surfaces plays a key role in mediating protein-target interactions. MaSIF (molecular surface interaction fingerprinting) couples a robust protein surface patch representation (encapsulating both geometrical and chemical information) with a polar convolutional neural network to create ‘molecular fingerprints’ that can be used with a downstream learning method for partner-independent interface prediction. Methods that rely solely on geometrical and chemical features can be expected to outperform conservation-reliant methods in cases where the multiple sequence alignments of the input proteins are shallow. Such predictors have the potential to exhibit superior performance to conservation-based predictors on tasks such as paratope-epitope binding prediction, where the complementarity determining regions (CDRs) of antibodies exhibit hypervariability (ed. JD Abbott et al., 2004).

We hypothesize that a method combining detailed geometrical descriptions of the input proteins and robust conservation-based features has the potential to exhibit state-of-the-art performance in both high and low homology realms. To this effect, we attempt to improve BIPSPI (Sanchez-Garcia *et al.*, 2019, 2022), a partner-specific protein interface predictor, with geometrical and chemical descriptions of protein surfaces extracted using MaSIF’s (Gainza *et al.*, 2020) feature extraction protocols. The MaSIF framework constructs a mesh of the protein’s surface, embedded with geometrical and chemical information, which is subsequently decomposed to form protein surface patches. We augment BIPSPI’s existing residue-pair vector representation with accurately mapped surface patches and ascertain whether such patches are meaningful in the context of partner-specific protein interface prediction by performing performance comparisons on Docking Benchmark v5.5 (Vreven *et al.*, 2015) with a variety of metrics. We also compare the relative impact of BIPSPI’s existing feature set and the features added by MaSIF using feature importance computations.

We acknowledge that, while highly informative, training and testing with patch-level data can be computationally expensive, both in terms of memory usage and operation time. Thus, we also develop unsupervised learning-based protein surface patch compression methods to significantly reduce the size of protein surface patches to tractable levels that allow for reduced memory consumption and increased computation speed during training, and the feasible large-scale storage of patches.

2 Methods

2.1 Improving BIPSPI using protein surface patches extracted using MaSIF

2.1.1 Problem overview

The specific problem under consideration is the partner-specific protein-protein interface prediction problem that can be stated as 'Given proteins A and B bind and their unbound structures, predict whether residue r_a in protein A interacts with residue r_b in protein B upon the formation of the complex A-B". We approach this problem from a supervised learning perspective, where a machine learning algorithm is trained on a dataset constituted of residue-residue feature vectors pooled from a set of non-redundant protein-protein complexes. The residue-residue feature vectors are vectors constituted of numerical representations of the properties representing residue-pairs of the form (r_a, r_b) from proteins A and B, respectively.

Feature computation is performed on the unbound structures of proteins A and B. We define a cross-protein pair of residues as being 'interacting' if they occur within 6Å of each other in the bound form of the complex (Minhas *et al.*, 2014; Sanchez-Garcia *et al.*, 2019). Residues in the bound forms of the interacting proteins are mapped to their counterparts in the unbound forms using a combination of structure and sequence-based alignments. Labels are assigned to cross-protein residue-pairs using residue-residue distance computations performed using the bound structures of proteins A and B. Figure 1 illustrates the data processing workflow associated with the problem.

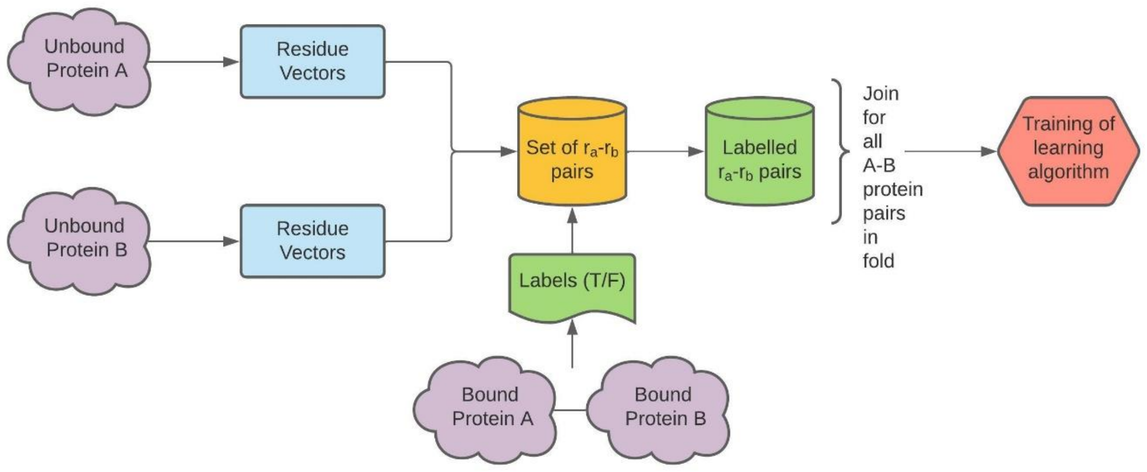

Figure 1: Flowchart elucidating the workflow associated with residue-residue contact prediction

We train the model on unbound structures to best simulate what is expected to be received at the time of practical use. While it is, in principle, possible to train on bound structures, we believe that models trained on unbound structures have greater potential to capture the effects conformational change can have at interfaces.

The objective of this study is to modify BIPSPI, a partner-specific protein interface predictor, to utilize geometrical and chemical features computed using MaSIF, a framework for the extraction of meaningful features from the surfaces of proteins. We compare BIPSPI's default performance to that achieved with the use of the additional surface-patch-based features. We compare a total of six models with varying sets of features to ascertain whether the addition of surface-derived structural features offers a significant improvement over the existing framework.

To supplement the potential use of patches for protein interface prediction and other tasks in data-driven protein biology, we also develop a compression framework that significantly reduces the size of protein surface patches for reduced memory consumption during model training and for long-term storage.

2.1.2 Dataset

For the purpose of training and testing our model, we use Docking Benchmark v5.5 (Guest *et al.*, 2021). The dataset consists of 257 binary non-redundant protein-protein complexes.

The dataset is structurally non-redundant in that no two complexes in the dataset have interacting domains that belong to the same SCOP family (Lo Conte *et al.*, 2000). Unlike other protein-protein interaction datasets, which solely contain bound forms of complexes, Docking Benchmark v5.5 contains the structures of both the bound and unbound forms of proteins in binary complexes. This allows for machine learning algorithms to receive information that is closer to what is observed at test time. Each complex is represented by a set of four PDB files: the bound and unbound forms of the 'ligand' protein and the bound and unbound forms of the 'receptor' protein. The dataset has been used extensively in the literature to train and test machine learning models for protein-protein interaction prediction, allowing us to potentially benchmark the modified model's performance against other predictors in the literature in the future.

We exclude complexes that are unable to be processed by the MaSIF framework, resulting in the use of a subset of 192 binary protein-protein complexes. During training time, the data obtained from each complex is such that the ratio of positive to negative samples is 1:2.

2.1.3 Overview of BIPSPI

BIPSPI is a highly competent partner-specific protein interface predictor described in (Sanchez-Garcia *et al.*, 2019). BIPSPI stands for 'xgBoost Interface Prediction of Specific-Partner Interactions'. The method was trained to predict the interfaces of protein-protein complexes using features derived from the sequences and/or structures of interacting proteins. It is a tractably modifiable algorithm that can be trained in reasonable time-frames and is near state-of-the-art in terms of performance.

Figure 2 illustrates the steps involved in the BIPSPI workflow. The first step in BIPSPI is a residue-pair interface classification task using a combination of sequence-derived and structure-derived features. In the second stage of classification, a second predictor is provided with the classification scores obtained by the first predictor and newly computed pairwise environment prediction scores alongside the original set of features. The predictions made by the second classifier are subsequently used to determine the interface of the complex.

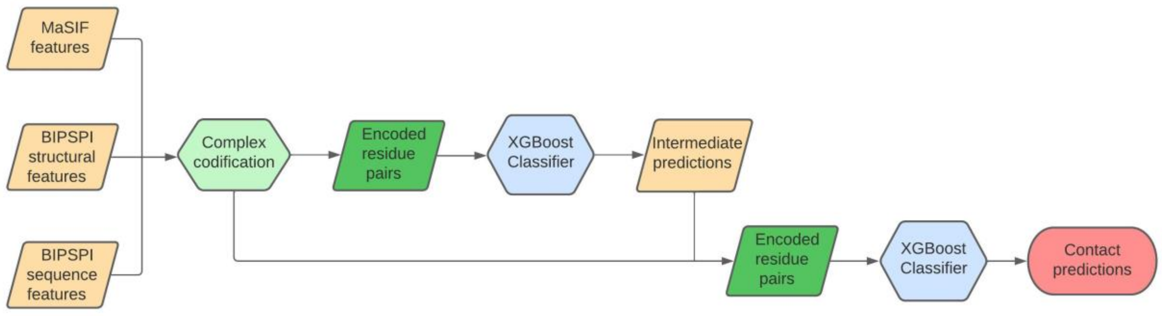

Figure 2: Illustration of the steps involved in residue-residue contact prediction using BIPSPI

Both predictors used in BIPSPI are XGBoost (Chen and Guestrin, 2016) classifiers. XGBoost is a gradient-boosted decision tree framework that has been shown to be exceptionally competent at a wide range of classification tasks that utilize tabular forms of data representation (Chen and Guestrin, 2016). Gradient boosting entails the creation of an ensemble of base learners constructed additively, such that each successive base learner rectifies the shortcomings of the set of base learners constructed in the previous iterations (Friedman, 2001).

In gradient boosting methods, predictive models are built progressively over a set number of training iterations. At each step, the negative gradient is computed using the existing model at that iteration, the input variables, and the loss function of choice. The negative gradients subsequently become target variables – a base learner is trained to predict the set of negative gradients from the input variables. Subsequently, a step size to update the model is chosen in accordance with the gradients computed. The model is then updated by adding the base learner to the existing set of learners. In gradient-boosted decision tree frameworks such as XGBoost, the base learners are decision trees. The below algorithm presents this in a logical format:

Algorithm: General Gradient Boosting

```
1. Initialize the prediction function
2. For index in range(1, M):
       a. Compute negative gradients (pseudo-residuals) with respect to the
          existing prediction function
```

   b. Fit a new base-learner (regression tree) to the pseudo residuals computed in (a)
   c. Identify an appropriate multiplying factor for each of the terminal nodes of the tree constructed in (b)
   d. Update the existing function with the new base-learner constructed
3. Output the function obtained after the M^th iteration

The general gradient boosting algorithm described above can be easily adapted for classification with an appropriate choice of loss function. For binary classification, the loss used is the log loss function described below, where y and p represent the ground truth and $P(y=1)$, respectively:

$$L = -(y\log(p) + (1-y)\log(1-p))$$

Gradient boosting algorithms are also amenable to regularization. Regularization refers to the process by which a model is modified to increase its capacity to generalize. As a means of regularization via shrinkage, the base learner constructed after 2.c in the above algorithm is scaled with a learning rate factor ν (where $0 < \nu < 1$) prior to addition to the existing model (Friedman, 2001). The use of the shrinkage parameter reduces overfitting by scaling down the influence of each added base learner. Regularization can also be performed by restricting the number of iterations.

XGBoost improves on the general gradient boosting algorithm by using a sparsity-aware learning algorithm, a weighted quantile sketch to efficiently compute tree split proposal, and adds other computational optimizations that improve the method's scalability. A detailed mathematical explanation of the algorithmic advancements made by XGBoost over general gradient boosting is outside the scope of this thesis and is best explained in (Chen and Guestrin, 2016).

2.1.4 Sequence-derived features used by BIPSPI

Most of BIPSPI's sequence-derived features are conservation-related. The conservation-derived features that BIPSPI uses are computed from multiple sequence alignments performed on the UniRef90 (Suzek *et al.*, 2007) database using PSI-BLAST (Altschul, 1997). UniRef90 is a non-redundant sequence database maintained by the UniProt consortium, constructed by clustering UniRef100 at 90%

sequence identity. Three iterations of PSI-BLAST are performed, and the PSSMs and PSFMs extracted from the final iteration are used. In addition to PSSMs and PSFMs, the multiple sequence alignments produced by PSI-BLAST are analyzed using AL2CO (Pei and Grishin, 2001) to compute an estimated conservation score.

The PSFM profile represents the frequency with which each of the twenty amino acids is found at a specific position across the sequences considered in the multiple sequence alignment. The PSSM profile measures the log-likelihood of finding each of the twenty amino acids at that specific position against the background distribution of amino acids. For a given residue, in addition to its specific PSSM and PSFM profiles, the PSSM and PSFM profiles of a window of amino acids (5 residues to each side) are also used in its representation. The information contained at each position is also used to represent residues.

In addition to these features, the one-hot encodings of the pair of residues along with those of residues in their corresponding aforementioned windows are used in the residue-pair representation.

2.1.5 Structure-derived features used by BIPSPI

Alongside sequence-based features, BIPSPI uses an extensive array of structural features at prediction time. The structural features are computed from the unbound structures of the interacting complexes using PSAIA (Mihel *et al.*, 2008), DSSP (Kabsch and Sander, 1983), and Biopython (Cock *et al.*, 2009). Most of the structural features used in BIPSPI are computed at the level of individual residues.

The key structural features encoded in BIPSPI's residue-pair representation are the following:

Accessible Surface Area (ASA) or Accessibility: The term refers to the area of the residue that is accessible to solvent molecules (Lee and Richards, 1971). It is measured in squared Angstroms and is computed by PSAIA using the rolling ball algorithm (Shrake and Rupley, 1973). Figure 3 depicts the accessible surface area associated with a macromolecule. The method also computes the relative accessible surface area (RASA) (Tien *et al.*, 2013) for each residue, which is a ratio of the solvent-accessible surface area to the maximum possible solvent-accessible surface area for that amino acid. BIPSPI uses the total, backbone, side-chain, polar, and non-polar ASA and RASA values assigned to each residue.

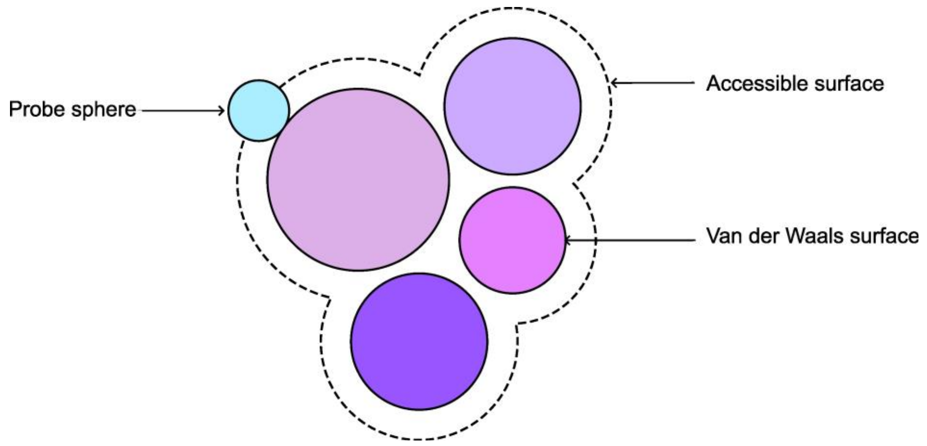

Figure 3: Accessible surface area associated with a macromolecule. A sphere of fixed radius is used to probe the surface of the macromolecule.

Half-sphere Exposure (HSE): Half-sphere exposure (Hamelryck, 2005) is computed using the Biopython package considering a radius of 12Å. HSE contains two components: HSE-up and HSE-down. For a given residue, these components represent the number of neighbours contained within the upper and lower halves of the sphere defined by a radius of 12Å, the C_α atom, and the plane perpendicular to the plane containing the C_α-C_β vector (Figure 4).

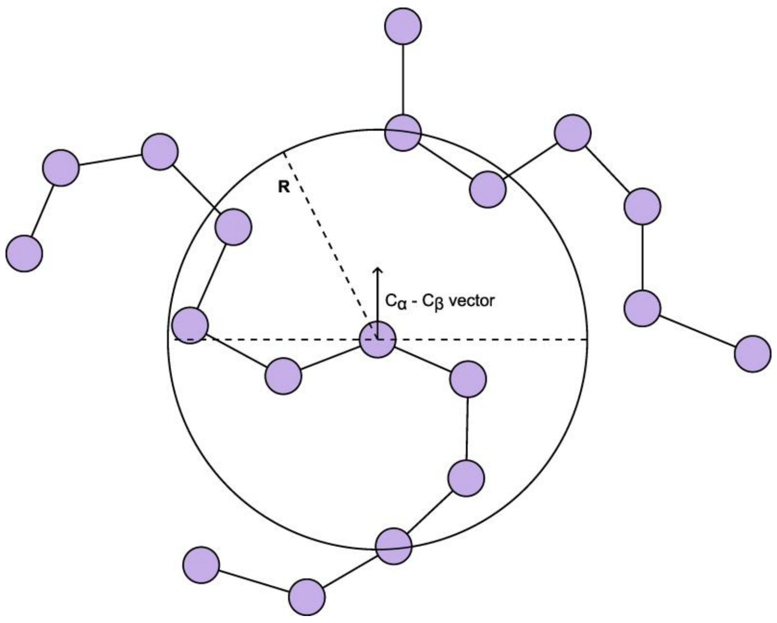

Figure 4: Half-sphere exposure computation. For a set radius R, the number of neighbours contained in the upper and lower hemispheres in the figure correspond to HSE-up and HSE-down respectively.

Secondary Structure: Each residue is assigned an 8-dimensional vector depending on the secondary structure element it is a part of in the 3D structure of the protein. The secondary structure label for a given amino acid is computed using DSSP (Kabsch and Sander, 1983).

Depth Index (DPX) (Pintar *et al.*, 2003): The depth index of a given residue is defined as the smallest distance from any of its atoms to the closest solvent-accessible atom of the protein.

Protrusion Index (CX): This quantity measures the extent of protrusion exhibited by atoms of the protein. Protrusion is computed by PSAIA using the CX algorithm (Pintar *et al.*, 2002). For both depth index and protrusion index, the total mean and standard deviation, the side chain mean and standard deviation, and the maximum and minimum values for a given residue are used.

2.1.6 Residue neighbourhood codification

Each residue is also assigned a structural environment feature vector summarizing the value of each of its features within its local structural environment. Two residues are considered neighbours if they are connected by an edge in the protein's Voronoi diagram defined by its C_α atoms. The neighbourhood radius used to construct the Voronoi diagram is 30Å.

The second stage of classification uses the predicted interaction scores computed by the first XGBoost classifier. For a given pair of residues, a pairwise environment score is also calculated using its neighbours as defined by the proteins' Voronoi diagrams. A comprehensive description of the neighbourhood codification is available in (Sanchez-Garcia *et al.*, 2019).

2.1.7 Overview of MaSIF

MaSIF (Gainza *et al.*, 2020) is a framework developed as a proof-of-concept for the postulation that protein surfaces contain informative geometrical and chemical patterns that mediate intermolecular interactions and that these patterns, while not directly perceivable to the eye, can be learnt by a machine learning framework and used for various tasks such as interface site prediction and pocket classification. The method uses polar convolutional neural networks in conjunction with ‘patches’ (that encapsulate local geometrical and chemical patterns) extracted from the molecular surfaces of proteins. Owing to its non-reliance on conservation-derived features such as PSSMs and PSFMs, MaSIF is particularly powerful in low sequence co-homology realms. In its existing form, while MaSIF has been proven an effective partner-independent interface site predictor, the method has not been employed specifically for the purpose of partner-specific protein-protein interface prediction. It is also to be noted that the MaSIF methods were trained using complexes co-crystallized in the bound state and not on features derived from the unbound forms of these complexes.

In the context of improving the performance of BIPSPI, we are specifically interested in the pre-processing protocols MaSIF employs to extract data from the surfaces of interacting proteins. Since the features extracted using MaSIF do not explicitly draw from properties of the proteins, such as the folds of the various domains or the sequence (unlike BIPSPI), we believe that the sets of structural features encoded by the two methods are likely to be complementary. We use MaSIF’s feature extraction protocol to generate the molecular surfaces of the proteins in our dataset and perform feature computation for the vertices that constitute these surfaces. MaSIF describes each vertex on the surface of the protein using a set of five features (two geometrical features and three chemical features). Feature computation is performed at the level of the surface, following which it is decomposed into patches of set geodesic radius.

The protocol first re-protonates all proteins in the dataset using reduce (Word *et al.*, 1999) and generates triangular molecular surface meshes from them using MSMS (Sanner *et al.*, 1996). Following a regularization of the mesh, geometric and chemical features are assigned to the vertices that constitute the mesh. The features used have been described below.

2.1.8 Geometric features used by MaSIF

For each vertex on the mesh, MaSIF computes ‘Shape Index’ and ‘Distance-dependent curvature’.

Shape Index

The shape index (Gainza *et al.*, 2020) of a vertex numerically represents the local geometry around it, with values extending from -1 to 1. The shape index of a specific vertex remains constant across the several patches it can be a part of. It is defined as follows, where κ_1, κ_2, $\kappa_1 \geq \kappa_2$ are the principal curvatures:

$$\text{Shape Index} = \frac{2}{\pi} \tan^{-1} \frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}$$

Distance-dependent curvature

The distance-dependent curvature (DDC) (Yin *et al.*, 2009) of a vertex is computed at the level of each patch. Distance-dependent curvature ranges between [-0.7, 0.7], and for a given vertex, quantifies the relationship between the distance from that vertex to the central vertex of the patch and the surface normals of the two vertices. The DDC of a given vertex varies across the patches it is a patch of.

Shape Index and Distance-dependent curvature together encapsulate the local geometrical neighbourhood of a vertex. A patch consisting of several vertices can thus be expected to represent the topographical properties associated with an area on the surface of the protein. The propensity of a residue to be a part of the complex interface depends not only on geometric properties intrinsic to it but also on those of other residues in its vicinity.

2.1.9 Chemical features

MaSIF computes a set of three chemical features on the protein’s surface mesh. These are charge via Poisson-Boltzmann continuum electrostatics, free electron/proton donor capacity, and hydrophobicity.

MaSIF uses the Adaptive Poisson-Boltzmann Solver (APBS) (Jurrus *et al.*, 2018) suite to perform Poisson-Boltzmann continuum electrostatics computations. Each vertex is assigned a charge value in the range [-1, 1] after normalization. To determine the positions of free electrons/protons on the molecular surface, MaSIF uses a hydrogen-bond potential developed by Kortemme *et al*. (2003) as a reference, assigning to each vertex a value between -1 and 1, depending on the potential for it to be a bond acceptor or donor respectively (Kortemme *et al.*, 2003).

Finally, each vertex is assigned the hydrophobicity score of the amino acid that is the closest to it.

After feature computation, the mesh is decomposed into patches of geodesic radius 9Å. Each patch is defined by a central vertex. A geodesic is intuitively the shortest curve between two points along a surface. Since the protein surface representation generated is a discrete triangular mesh (that can be interpreted as a weighted graph), approximate geodesics are computed for vertices (relative to the central vertex) using the Dijkstra algorithm for shortest path determination on graphs. The number of vertices in each geodesic patch is restricted to 100 to ensure that the learning algorithm receives input of fixed size.

For each protein, the feature computation steps of MaSIF outputs patch coordinates, the features of each vertex in a patch, and index lists denoting the vertices that belong to each patch. We leverage this information to map patches produced by MaSIF to the corresponding residue feature vectors generated by BIPSPI.

Table 1 contains the full set of features we used across all tested models. With the exception of 'Previous step predictions', the values shown represent the number of variables associated with each feature per residue.

Table 1: Features used in the training of XGBoost models. Features annotated (B) are associated with BIPSPI, whereas those marked (M) are computed via MaSIF. The values shown represent the number of variables corresponding to each feature associated with each individual residue in a pair.

Feature Name	**Computation Method**	**No: of variables**
One-hot encoded amino acid symbol in sliding window and structural environment (B)	BIPSPI script	264
Conservation (B)	PSI-BLAST AL2CO	662
Solvent accessibility (B)	PSAIA	50
Hydrophobicity (B)	PSAIA	5
Depth index (B)	PSAIA	30

Protrusion index (B)	PSAIA	30
Secondary structure (B)	DSSP	16
Half-sphere exposure (B)	Biopython	35
Shape Index (M)	MaSIF script	100
Distance-dependent curvature (M)	MaSIF script	100
Charge (M)	MaSIF script	100
H-bond potential (M)	MaSIF script	100
Hydrophobicity (M)	MaSIF script	100
Previous step predictions	BIPSPI script	26

2.1.10 Mapping procedure

The number of vertices far exceeds the number of residues present at the surface of proteins – this implies that patches do not necessarily map one-to-one with residues. Since BIPSPI performs predictions at the level of residue-pairs and uses residue-level features, each residue needs to be mapped to the vertex closest to it on the surface of the protein. Using the coordinates of each vertex and the atomic coordinates of the protein, we use the nearest neighbour algorithm implemented for k-dimensional trees in SciPy (Virtanen *et al.*, 2020) to perform this operation in an efficient manner. Figure 5 represents the residue to vertex mapping. While it is hypothetically possible to map each residue to its n closest vertices, we decided against doing so as patches corresponding to proximal vertices can be expected to be highly redundant.

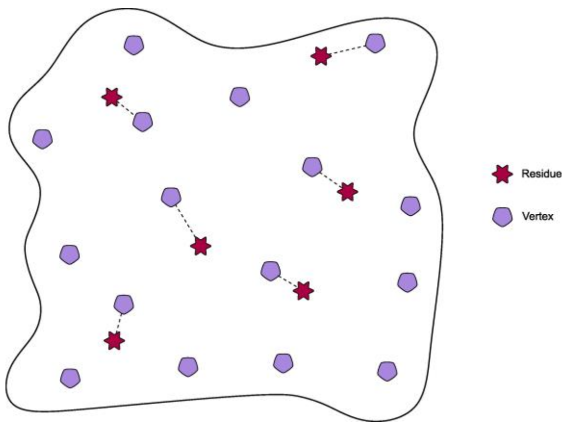

Figure 5: Pictorial representation of the residue to vertex mapping. Each residue is mapped to the vertex closest to it on the surface of the protein.

While it is possible to map every residue in a protein to a patch on its surface, it might not make sense to perform this procedure in cases where the residue in question is deeply buried. To determine the fraction of residues in our proteins for which this condition would be applicable, we construct the inverse map connecting each vertex to its closest residue. Subsequently, we identify the set of residues that are not the closest to any vertex on the surface and compute the mean percentage of residues affected over ligand and receptor complexes.

From Figure 6, we observe that, on average, only 11.43% of ligand residues and 15.71% of receptor residues are affected by this criteria. Since the number of residues accounts for such a small fraction of the total number of residues, we believe they are unlikely to significantly influence the accuracy of our model if included in the training set.

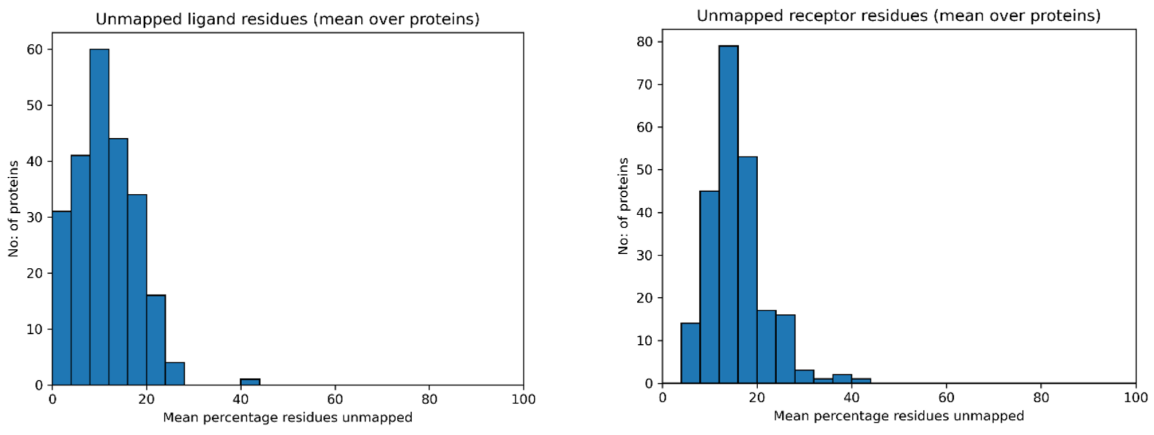

Figure 6: Distribution of the mean percentage of residues unmapped in ligand and receptor proteins respectively

2.1.11 Patch 'sorting'

Since XGBoost accepts only linear inputs, patches are linearized prior to use for training. Different rotations of a given patch correspond to different possible linearizations – this implies that the same area on the surface of the protein can result in multiple distinct numerical representations. To incorporate a degree of rotation invariance, we sort each patch along the five feature axes in order of magnitude prior to linearization. While we cognize that this procedure corrupts the inherent geometry of a patch, it serves to significantly reduce the amount of variance seen at a given index across all patches. Sorting a patch fixes the ‘meaning’ associated with each position of its linear representation, potentially resulting in better performance when used with frameworks such as XGBoost that only accept linear input representations.

2.1.12 Variants of BIPSPI considered

We compare a total of six models described below:

BIPSPI-seq: BIPSPI-seq uses only features that can be derived from the amino acid sequences of the chains of the interacting pairs of proteins.

BIPSPI-default: BIPSPI-default, as the name suggests, is the default version of BIPSPI that uses both sequence-derived and structure-derived features.

BIPSPI-patch: This model replaces the structural features used by BIPSPI with the surface patch-level features computed by MaSIF. It contains all the sequence-based features used in BIPSPI-seq.

BIPSPI-default-patch: This model contains all features used in BIPSPI-default and includes surface patch-level features extracted using MaSIF in its residue-pair representation. BIPSPI-default-patch is expected to exhibit performance either equal or superior to that of BIPSPI-default.

BIPSPI-patch-sorted: This model differs from BIPSPI-patch in its use of sorted patches instead of directly linearized patches.

BIPSPI-default-patch-sorted: This model differs from BIPSPI-default-patch in its use of sorted patches instead of directly linearized patches.

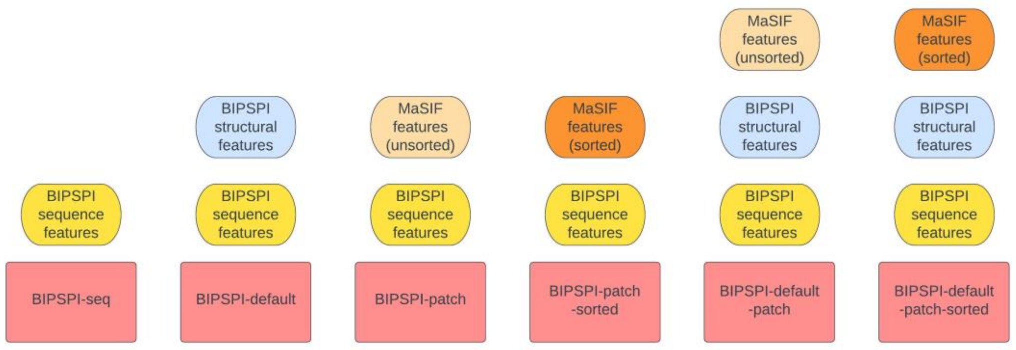

Figure 7: Feature sets associated with the six models under consideration

2.1.13 Evaluation

The following evaluation metrics are used to compare the various models under consideration:

ROC-AUC: The receiver operating characteristic curve (ROC curve) is a plot that describes the relationship between a model's binary classification performance and the decision threshold chosen. Each point on the curve represents the True Positive Rate and False Positive Rate of the classifier computed at that specific decision threshold. The area under this curve (AUC) is representative of the model's performance across a range of different decision thresholds. A random classifier is expected to have an ROC-AUC of 0.5, with a perfect predictor exhibiting a value of 1.

Precision (PRE): At a given decision threshold, a model's precision is the ratio of the number of true positives to the total number of samples predicted as positive.

$$\mathrm{PRE} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

Recall (REC) (or True Positive Rate): At a given decision threshold, a model's recall is the ratio of the number of true positives to the total number of positive samples contained in the dataset.

$$\mathrm{REC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Accuracy (ACC): At a given decision threshold, accuracy is the ratio of the number of correctly classified cases to the total number of cases.

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP} + \mathrm{FN}}$$

Negative Predictive Value (NPV): At a given decision threshold, the negative predictive value of a classifier is the ratio of the number of true negatives to the total number of instances predicted by it as negative.

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}}$$

Specificity (SPC): At a given decision threshold, specificity is the ratio of the number of true negatives to the total number of negative instances contained in the dataset.

$$\mathrm{SPC} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

False Positive Rate (FPR): At a given decision threshold, the false positive rate is the ratio of false positives predicted to the total number of negative instances contained in the dataset.

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

Matthews Correlation Coefficient (MCC): This is a metric commonly used to evaluate the performance of models trained to perform prediction on highly imbalanced data. MCC is defined as follows (where TP, TN, FP, and FN stand for True Positive, True Negative, False Positive, and False Negative, respectively):

$$\mathrm{MCC} = \frac{\mathrm{TP} \cdot \mathrm{TN} - \mathrm{FP} \cdot \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP}) \cdot (\mathrm{TP} + \mathrm{FN}) \cdot (\mathrm{TN} + \mathrm{FP}) \cdot (\mathrm{TN} + \mathrm{FN})}}$$

The performances of the six models are evaluated using 10-fold cross-validation at the level of protein complexes. The dataset is randomly split into 10 ‘folds’ containing 19 complexes each. The model is trained on nine folds and is evaluated on the excluded fold. This process is repeated, sequentially leaving out all ten folds. The model’s performance with respect to predicting the interface of a given complex is computed when it is tested on the fold containing the complex. The model’s overall performance with respect to a given evaluation metric is the mean of its performance with respect to the metric across all 192 complexes. All six models were trained with the same ten folds.

2.1.14 Feature Importances

In addition to the performance metrics for the six models, we also compute the feature importances associated with each of the input features used to train the model. Feature importances have been computed using Gain.

The ‘Gain’ associated with a feature refers to the reduction in loss incurred at a split involving that feature in a tree (that is a part of the overall XGBoost model). The total gain associated with a feature is the sum of its gain values across all trees it is a part of. We compute the relative gain associated with each feature by ascertaining the ratio between the total gain associated with it and the total gain accumulated across all features used. The expression used to compute the gain at a particular split of the tree is the following (Chen and Guestrin, 2016):

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right] - \gamma$$

As per XGBoost documentation, the first term quantifies the score on the left leaf of a split, the second denotes the score on the right leaf at a split, the third term represents the score at the leaf prior to the split, and the fourth term represents the regularization associated with the added leaf.

For each feature used in our models, we compute its global importance and mean importance per variable. As evident from Table 1, each feature corresponds to a set of variables. The global importance measures a feature’s total impact on the model’s performance. We interpret the mean importance per variable as a measure of how efficient a feature is at improving the model’s performance in relation to the number of variables that constitute it. For example, a feature that has a large contribution to the total gain and is constituted of a small number of variables can be expected to have a high mean importance per variable.

2.2 Data-driven compression methods for protein surface patches

While protein surface patches represent a rich source of information for various prediction tasks, such as interface site prediction and ligand binding prediction, they also occupy a significant amount of space. For a patch radius of 9Å, MaSIF produces patches constituted of 100 vertices, with each vertex represented by five dimensions. The surface mesh of each protein in our dataset contains several thousand vertices. We develop a data-driven compression method to reduce the size of computed patches by a significant factor. The reduction of the sizes of protein patches allows for the following:

1) Increase in computation speed and reduction in memory consumption during the training of methods that utilize patches

For a constant input feature size, reducing the size of individual features while retaining the same amount of information can allow for the use of a greater number of distinct informative features, potentially improving model performance. The use of compressed informative features can also greatly facilitate the training of highly performant models in low computing capacity realms.

2) Feasible local storage of large protein surface information datasets

Pre-computed datasets of protein surface patches can greatly accelerate inference speed at test time. The compression of patches can significantly increase the amount of protein surface data that can be stored locally and allow for fast test-time inference by precluding the need for surface computation.

To accomplish this, we develop i) a principal component analysis-based (Pearson, 1901) method and ii) an autoencoder-based (Kramer, 1992) method of patch compression. Both methods described fall under the category of unsupervised learning methods. The data-driven models were developed and tested on a non-redundant dataset whose construction has been described in detail below.

2.2.1 Dataset construction

The dataset constructed to train the protein surface patch compression methods was designed to be non-redundant and disjoint from Docking Benchmark v5.5 to allow for the future use of the compression method with BIPSPI.

A cull was performed on structures available in the Protein Data Bank using PISCES (Wang and Dunbrack, 2003) to select PDBs whose structures were determined using X-ray Crystallography. The structures selected were such that their resolutions were superior to 2.0Å, and they exhibited R-factor values lower than 0.25. The sequence similarity criteria for culling was set to 5% to minimize redundancy in the dataset - no pair of structures in the dataset exhibited sequence-similarity greater than 5%.

The cull retrieved 2532 PDB files. The extracted PDB files were subsequently processed using MaSIF's pre-processing pipeline. Surface meshes were computed for all proteins in the dataset. This was subsequently followed by the computation of chemical and geometrical features. The patches were sampled from the patch datasets of individual proteins such that no two patches extracted from the same protein shared more than 20 vertices. We hoped to maximize the percentage of surface data utilized from a given protein while simultaneously minimizing the number of vertices shared between patches. The end result was a dataset consisting of 109,364 patches, each patch represented by 100 vertices described by five features per vertex.

2.2.2 Method I: Principal component analysis

Principal component analysis is a dimensionality reduction procedure commonly used to visualize high-dimensional data in 2 or 3-dimensions. 'Principal components' refer to basis vectors in high-dimensional space along the direction of maximum variance. Assuming that the input data is standardized (i.e., the values of the variables are scaled such that their mean is 0 and standard deviation is 1), the principal components represent the eigenvectors of the covariance matrix constructed from the data, in decreasing order of the variance explained. By construction, they are orthogonal linear combinations of the original input variables that maximize the variance of the samples projected on to them.

Given a data matrix containing n samples, described by m variables each, the following steps are involved in principal component analysis:

1. Standardize the data matrix: Scale each of the m variables in the dataset in accordance with its computed mean and standard deviation
2. Construct a covariance matrix from the standardized dataset: Given a standardized $n \times m$ data matrix $\mathbf{S}$, the covariance matrix $\mathbf{C}$ can be computed as $\mathbf{C} = \frac{1}{n-1}\mathbf{S}^*\mathbf{S}$
3. Compute the eigenvalues and eigenvectors of the covariance matrix
4. Sort the eigenvectors in order of decreasing eigenvalues
5. Choose a subset consisting of the first k eigenvectors to project the samples in the data matrix onto. These represent the basis vectors along the direction of maximum variance.

The procedure also allows for the computation of the percentage of variance explained by each principal component. The proportion of variance explained by a principal component is the ratio of its corresponding eigenvalue to the sum of all eigenvalues of the covariance matrix.

$$\text{Variance explained by } \mathrm{PC_i} = \frac{\lambda_i}{\lambda_1 + \cdots + \lambda_n}$$

The magnitude of this ‘explained variance ratio’ is indicative of how effective a principal component is at capturing the variance inherent to the dataset. As principal components are uncorrelated (i.e., orthogonal in space), the percentage variance accounted for by a subset of principal components can be computed through the direct addition of their explained variances.

To reduce the dimensionality of the original dataset, a set constituted of the first n principal components that captures a substantial proportion of the sample variance is constructed. The data is subsequently projected along the n principal components considered. If 1, 2, or 3 dimensions capture a significant fraction of sample variance, the data can be visualized in plots for researchers attempting to observe the relative spatial positioning of various datapoints. In this case, what fraction can be deemed 'substantial' is at the discretion of the researcher.

The patches of dimension [100, 5] contained in the sampled non-redundant patch dataset are linearized to 500 dimensions. The [109364, 500] dataset is then segmented into train, validation, and test sets of sizes 80000, 20000, and 9364, respectively. The training dataset is standardized along the axes of the variables to mean 0 and standard deviation 1. The standard deviation and mean for the 500 variables computed from the training dataset are used to scale the validation and test sets. The standardization procedure is performed to ensure that the magnitudes of the various features are comparable to not lend undue importance to certain features owing to their high absolute magnitudes. The number of components used to compute explained variance ratio and reconstruction loss was varied from 5 to 500 in increments of 5. The explained variance ratio and reconstruction loss were subsequently plotted against the number of components and have been included in Section 3.2.1.

2.2.3 Method II: Autoencoder

An autoencoder (Kramer, 1992) is an unsupervised learning architecture designed for the purpose of representation learning. In most practical use cases, an autoencoder consists of two families of functions A (parametrized by ϕ) and B (parametrized by ψ), where A transforms input from an n-dimensional real-valued representation to a p-dimensional latent representation, and B transforms the p-dimensional latent representation back to n-dimensions, where A and B minimize the expected "distortion" between the input and output representations (Bank *et al.*, 2021).

$$A_\phi : \mathbb{R}^n \to \mathbb{R}^p$$
$$B_\psi : \mathbb{R}^p \to \mathbb{R}^n$$

The most used reconstruction loss in autoencoders is squared error, defined as follows where x is the $input$ and $\tilde{x}$ is the reconstruction produced by the autoencoder:

$$r(x, \tilde{x}) = ||x - \tilde{x}||_2^2$$

When used in the context of data compression, an autoencoder learns from input data, the optimal functions A_ϕ and B_ψ to convert high-dimensional data of a given type to a low-dimensional latent space while retaining as much information contained in the original sample as possible. The appropriate ϕ and ψ are those that minimize the loss defined as:

$$L(\phi, \psi) = \frac{1}{N}\sum_{i=1}^{N} \left\| x_i - B_\psi\left(A_\phi(x_i)\right)\right\|_2^2$$

In practice, A and B are usually neural networks called the ‘encoder’ and ‘decoder’ - the loss minimization procedure is performed using gradient descent. It can be mathematically shown that if purely linear operations are used to construct an autoencoder with a single fully-connected hidden layer of size p, the weights trained span the same subspace as the one spanned by principal component analysis with p principal components when trained with the squared error loss function (Plaut, 2018). The advantage accorded to an autoencoder through the use of non-linear activations is the ability to learn a non-linear manifold. In practice, the encoder and decoder are both constituted of several hidden layers.

Autoencoders designed for the purpose of compression use a ‘bottleneck layer’. The bottleneck layer is generally the narrowest point of the autoencoder architecture and is conventionally the last layer of the encoder. The value of a bottleneck layer is two-fold in that 1) in the context of compression, it is directly responsible for dimensionality reduction, and 2) it prevents the network from overfitting and learning the identity function for the given dataset (Bank *et al.*, 2021). Intuitively, if the original n-dimensional input can be fully reconstructed by the decoder using the p-dimensional latent representation encoded by the encoder, the latter completely encapsulates the information contained in the former. Figure 8 depicts the structure of a compressional autoencoder.

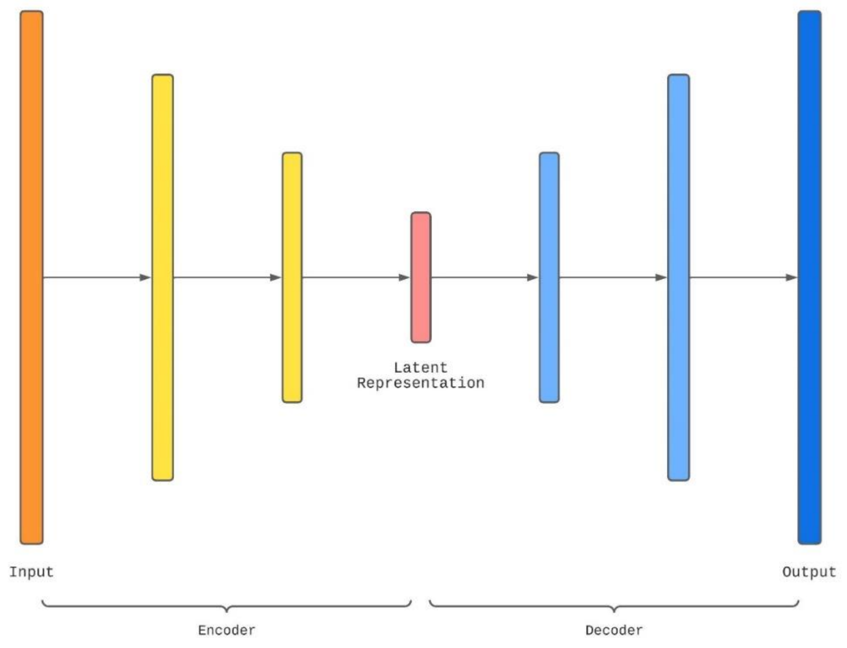

Figure 8: Diagram representing the structure of a compressional autoencoder

The protein patch dataset is segmented into train, validation, and test sets of sizes 80000, 20000, and 9364 patches, respectively. Similar to the protocol for principal component analysis, the training dataset was standardized to mean 0 and standard deviation 1. This removes the dependence of the reconstruction error on the magnitudes of the input variables. The patches in the validation set and test sets were linearized and transformed using the sample mean and standard deviation computed for the training set.

The reconstruction error used to train and validate the models is the Mean Squared Error loss function. Mean Squared Error (MSE) measures the average squared deviation between the true input and its reconstruction produced by the autoencoder across all the samples in the dataset. The best models during model selection are those with the lowest validation set MSE.

Though there are an infinite number of possible encoder and decoder structures that can be tested, we consider only symmetric architectures, where the number of layers in the encoder and decoder varies from 1-3, and the number of hidden units in each layer varies between 200, 300, and 400. Under these self-imposed constraints, we tested a set of 19 architectures each for autoencoders with bottleneck layers of size 50 and 100 dimensions, respectively. These constitute 10x and 5x reductions in the size of the 500-dimensional input vector, respectively. The autoencoder models are subsequently trained for 500 epochs with a batch size of 128 using the ‘Adam’ optimizer set to a learning rate of 0.0001. A single update to the model’s weights is carried out when its weights are altered based on the losses and corresponding gradients computed on a batch of 128 datapoints. An ‘epoch’ represents a full pass through the dataset – the same operation is performed using all remaining disjoint batches of size 128 constructed from the dataset. The training dataset is shuffled prior to each training epoch. ‘Adam’ is a method that adaptively alters the learning rate of the algorithm in response to the values of the computed gradients (Kingma and Ba, 2015).

The MSE losses are computed on the training and validation sets at the end of each epoch. Plots describing the evolution of these errors over 500 epochs are generated and are available in Section 3.2.1. As these curves can be noisy, the mean validation loss computed for the model over the final 50 epochs of training is used as a metric to compare the 19 models during model selection.

Similar to our hypothesis that the use of sorted patches could result in better classification performance, we believe that PCA-based and autoencoder-based compression methods trained on sorted patches could exhibit significantly improved MSE performance in comparison to those trained with geometrically-intact patches. We train the PCA and autoencoder-based approaches with both sorted and unsorted patches and report training and validation loss statistics.

3 Results and Discussion

3.1 Improving BIPSPI with protein surface patches extracted using MaSIF

3.1.1 Results

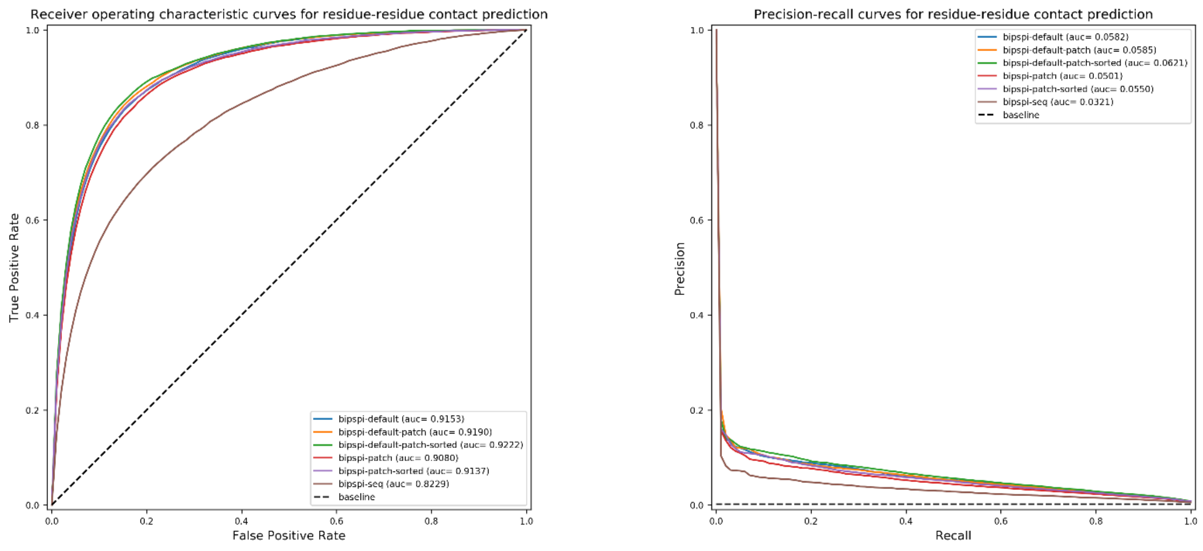

Figure 9: Receiver operating characteristic curves and precision-recall curves for all six models trained for the task of residue-residue contact prediction

Figure 9 illustrates the receiver operating characteristic curves and precision-recall curves of the six models tested. In addition to structural features, all models shown here use the same set of sequence-derived features. We observe that the model trained with structural features from both MaSIF and BIPSPI exhibits the best ROC-AUC performance across the six models that were tested. We also observe that for a given feature set, sorting the input surface patches appears to improve classification performance: the XGBoost models trained on sorted patches, *BIPSPI-default-patch-sorted* and *BIPSPI-patch-sorted*, perform better than their counterparts trained on unsorted patches, *BIPSPI-default-patch* and *BIPSPI-patch* respectively. The model trained solely with sequence-derived features and BIPSPI's default structural features (*BIPSPI-default*) exhibits marginally better performance than both models that utilize sequence-derived features and MaSIF's structural features (*BIPSPI-default-patch* and *BIPSPI-patch*). These trends are mirrored in the precision-recall curves plotted in Figure 9. The baseline in both plots represents the performance expected of a random classifier. We have also included the performance of the model trained only on sequence data (*BIPSPI-seq*) as an additional baseline to quantify the improvement in performance brought forth by the use of structure-derived features. The model trained only on sequence-based features exhibits performance considerably inferior to that of the remaining models.

Table 2: Performance summary of the six models evaluated using 10-fold cross-validation across nine performance metrics. All models were trained using the same ten folds.

Model Name	**ROC-AUC pooled**	**ROC-AUC mean**	**MCC**	**PRE**	**REC**	**ACC**	**FPR**	**SPC**	**NPV**
BIPSPI-default-patch-sorted	0.9333	0.9222	0.1052	0.0249	0.4866	0.9713	0.0280	0.9720	0.9992
BIPSPI-default-patch	0.9301	0.9190	0.1013	0.0225	0.5053	0.9670	0.0324	0.9676	0.9992
BIPSPI-patch-sorted	0.9257	0.9137	0.1006	0.0222	0.5054	0.9665	0.0328	0.9672	0.9992
BIPSPI-patch	0.9197	0.9080	0.0946	0.0227	0.4382	0.9714	0.0278	0.9722	0.9992
BIPSPI-default	0.9284	0.9153	0.0999	0.0216	0.5127	0.9648	0.0345	0.9655	0.9993
BIPSPI-seq	0.8463	0.8229	0.0644	0.0146	0.3385	0.9652	0.0339	0.9661	0.9990

Table 2 contains performance metrics computed for all six models using 10-fold cross-validation on the 192 complexes contained in our dataset. *BIPSPI-default-patch-sorted* exhibits the best performance across most computed metrics. Similar to what is observed in the receiver operating characteristic and precision-recall curves (Figure 9), for pairs of models trained with the same set of features, the variants trained on sorted patches exhibit better performance than their counterparts trained on unsorted patches. The values for metrics such as MCC, FPR, and ACC are computed by identifying a ‘best threshold’ that maximizes the performance of each of the classifiers. Two ROC-AUC scores are computed. ROC-AUC mean refers to the mean of the AUCs achieved on individual complexes during 10-fold cross-validation. ROC-AUC pooled represents the model’s AUC as computed from the results pooled from all complexes.

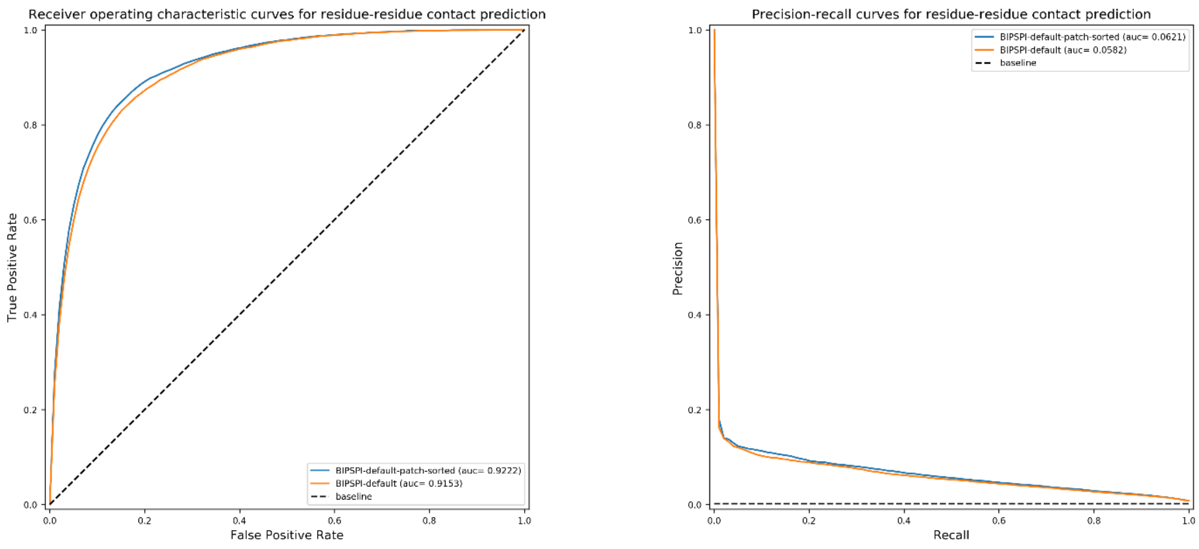

Figure 10: Receiver operating characteristic curves and precision-recall curves for *BIPSPI-default-patch-sorted* and *BIPSPI-default*

Figure 10 directly compares the receiver operating characteristic curves and precision-recall curves of the best-performing model (*BIPSPI-default-patch-sorted*) to those of the unmodified version of BIPSPI (*BIPSPI-default*). The addition of sorted patches generated by MaSIF to BIPSPI's structure-based feature set improves the ROC-AUC mean by 0.69 and PR-AUC by 0.0039.

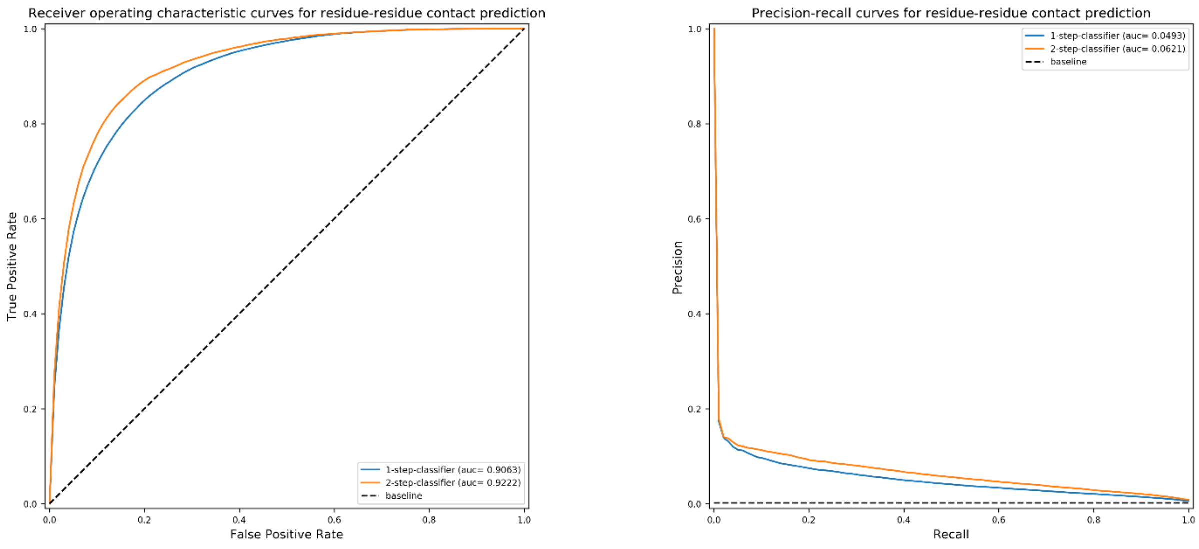

Figure 11: Receiver operating characteristic curves and precision-recall curves for 1-step and 2-step variants of *BIPSPI-default-patch-sorted*

From Figure 11, it is clear that there is a significant improvement in performance between the 1-step and 2-step variants of the model trained on structural features from both MaSIF (sorted) and BIPSPI. This trend is consistent across all six models.

In addition to the receiver operating characteristic and precision-recall curves, we also use XGBoost's in-built feature importance computation function to identify the features the algorithm considers to be the most important for performing predictions for all five models that use structural features. Feature importances were computed for the 1-step and 2-step variants of each of the five models that use structural features. Since each feature corresponds to a set of variables, we also compute the mean importance per variable of all features used by our models. We show both 'grouped' and 'comprehensive' plots for each of the five models that use structure-based features. In grouped plots, the structural features are grouped as either MaSIF features, BIPSPI features, Conservation features, or 'Other features' depending on their sources. Only 'amino acid identity' belongs to the 'Other features' category. The comprehensive plots represent the relative importance of *all* features used by the model as individual wedges on the pie chart. While the 2-step variants perform better than the 1-step variants, the feature importance plots for the former are heavily influenced by the predictions made in the first step and are hence less informative for the purpose of ascertaining the relative impacts of the various features used. As a result, we present only the feature importances computed for the 1-step classifiers in this section. Feature importance plots for the 2-step classifiers are available in the attached appendix.

Figures 12 and 13 represent the extent of importance the XGBoost model trained on the default set of BIPSPI features assigns to the various input features during training. Conservation appears to be the most influential of all the features, accounting for 64.98% of the total gain across all splits, and is almost four times as important as the second-most important feature (Accessibility), which accounts for 16.96%. Accessibility, however, appears to have the highest per-variable importance of all the features. Amino acid identities appear to be the least important to the model during training from the perspective of mean importance per variable. The feature importances computed for the model trained on the default set of features represent a control that the feature importances computed for the other models can be compared to.

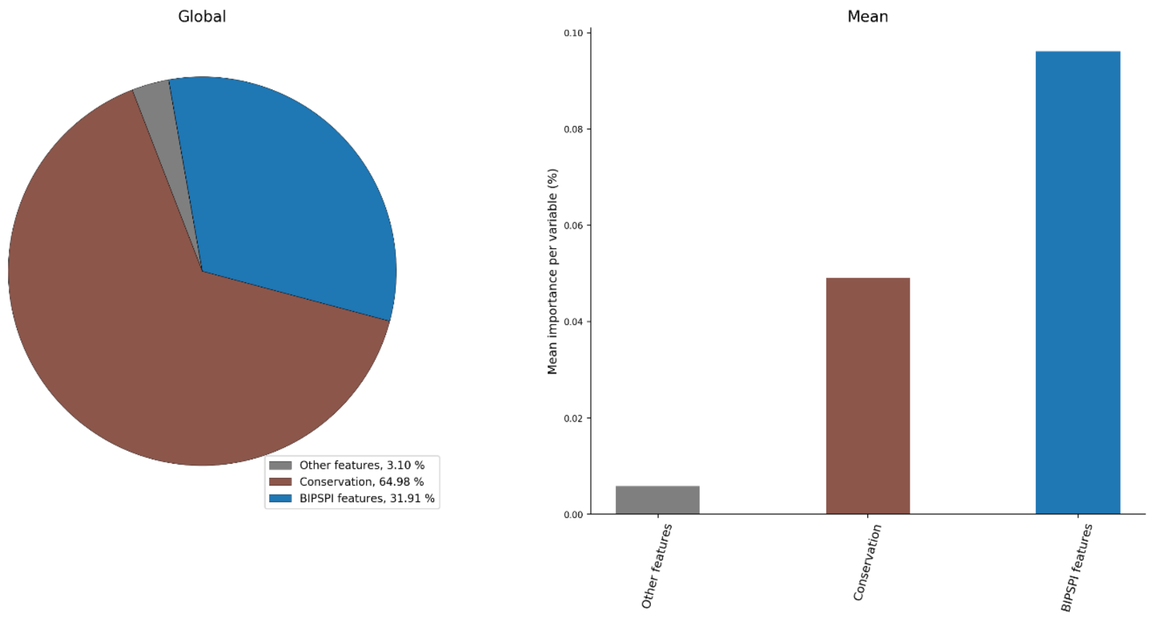

Figure 12: Global and mean feature importances associated with the features used in the training of *BIPSPI-default*. The structural features associated with BIPSPI are grouped together in a single category called ‘BIPSPI features’.

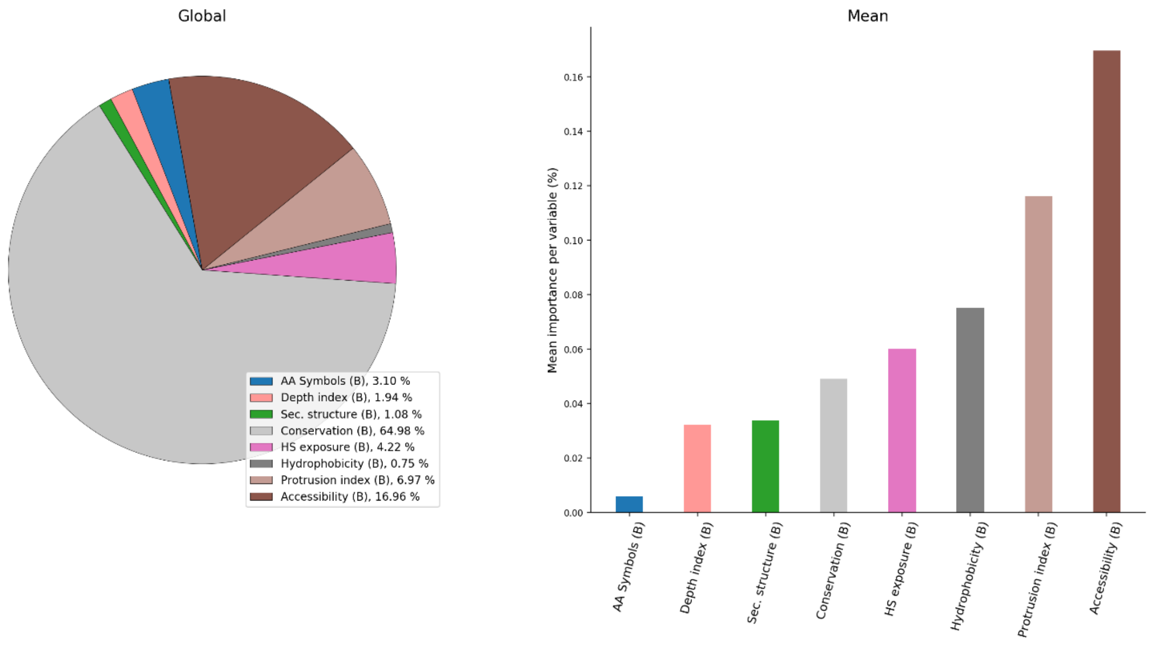

Figure 13: Global and mean feature importances associated with the features used in the training of *BIPSPI-default*

As is evident from Figures 14 and 15, when BIPSPI's structural features are replaced with patches (unsorted), the extent of influence held by conservation drops significantly from 64.98% to 56.98%. While conservation still is the most important type of feature, the extent of influence structural features have on the model has increased substantially in comparison to their proportions seen in *BIPSPI-default*. Distance-dependent curvature appears to be the most important of the five features contained in the patches, accounting for 12.42% of the total importance. Out of the three chemical features, free electron donor/acceptor potential (H-bond) has the least importance. It is to be noted that while the influence of structural features has increased compared to *BIPSPI-default*, the performance of the model is noticeably worse, clocking in at an ROC-AUC mean of 0.9080 in comparison to the 0.9153 exhibited by *BIPSPI-default*. The geometrical features appear to have higher global and mean importances than the chemical features in the patches.

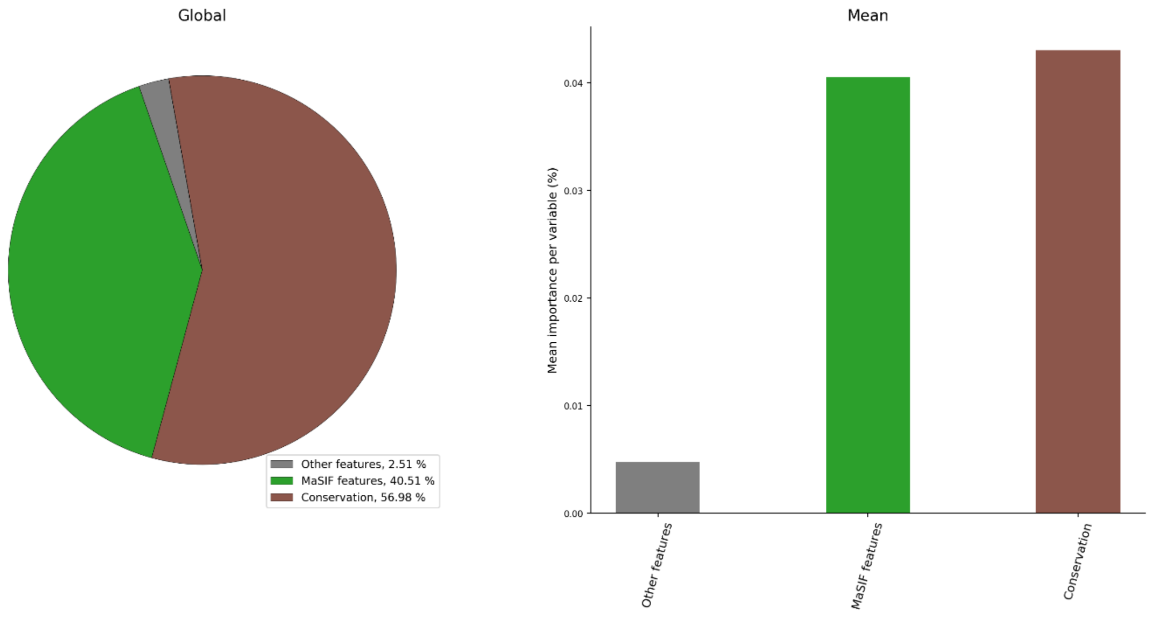

Figure 14: Global and mean feature importances associated with the features used in the training of *BIPSPI-patch*. The structural features associated with MaSIF are grouped together in a single category called 'MaSIF features'.

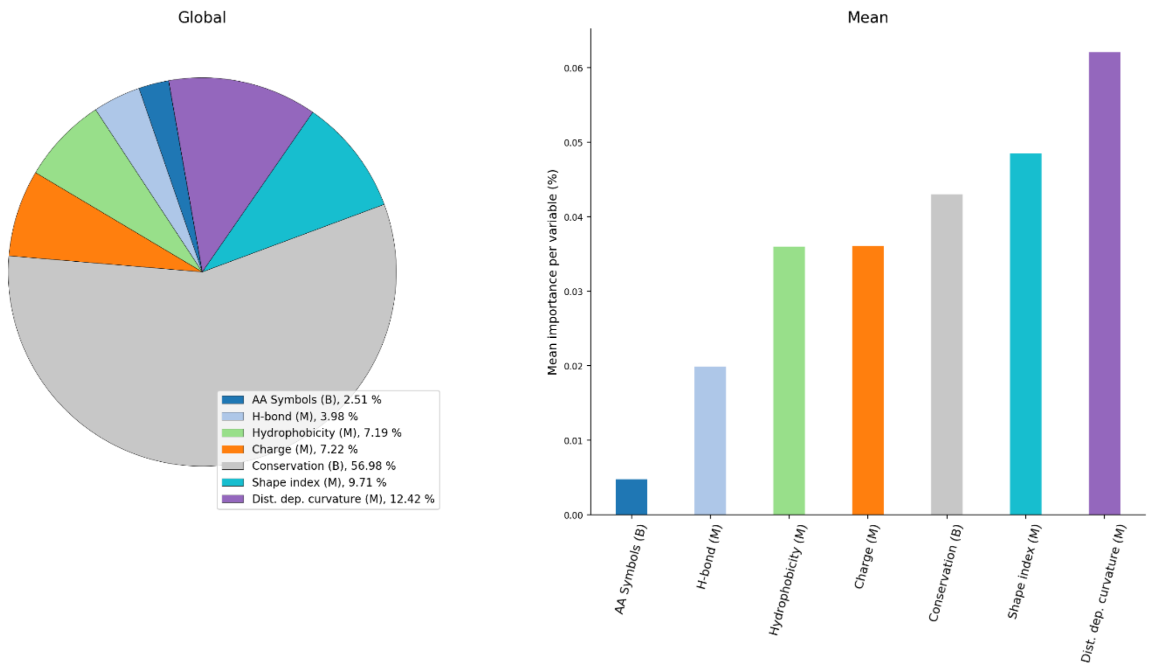

Figure 15: Global and mean feature importances associated with the features used in the training of *BIPSPI-patch*

Figures 16 and 17 show that supplying the model with sorted patches appears to improve performance while simultaneously reducing the reliance of the model on patch-based features during training. The extent of influence of sequence and structure-based features for *BIPSPI-patch-sorted* is comparable to that exhibited by *BIPSPI-default*. The conservation class of features regains its position as the most influential, surpassing its importance in the control model. Consistent with what was observed in *BIPSPI-patch*, distance-dependent curvature appears to be the most important MaSIF-derived feature during training, accounting for 7.19% of the total importance. The similarities continue with charge and hydrophobicity remaining comparable and free electron donor/acceptor potential being the least important feature. The performance of the model trained on sorted patches is superior to that of the model trained on unsorted patches.

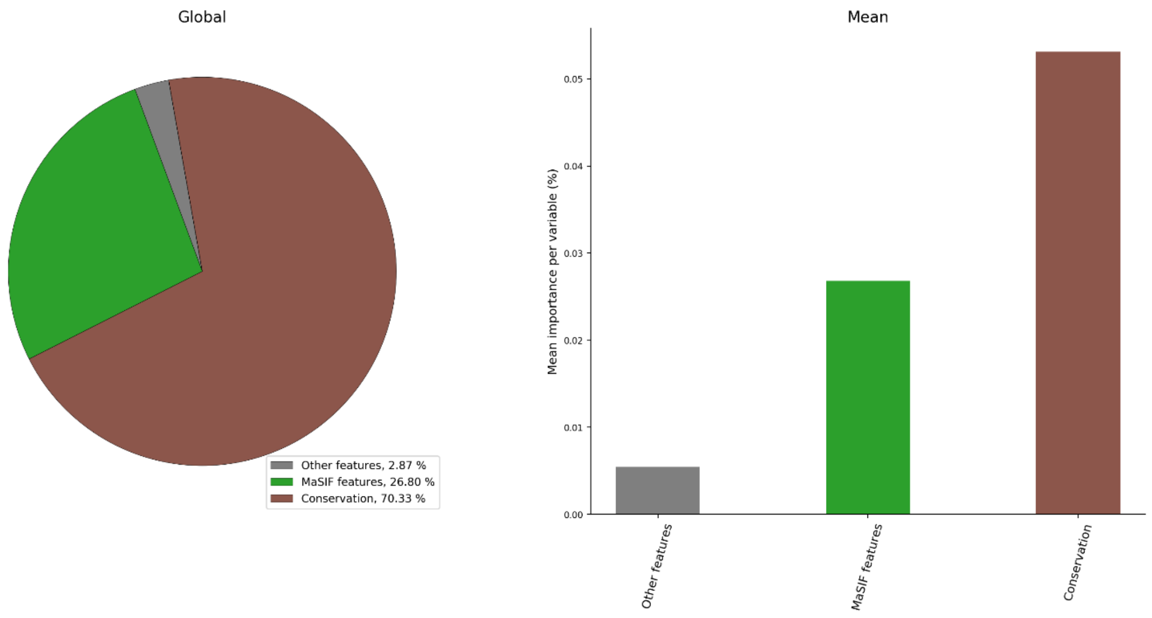

Figure 16: Global and mean feature importances associated with the features used in the training of *BIPSPI-patch-sorted*. The structural features associated with MaSIF are grouped together in a single category called ‘MaSIF features’.

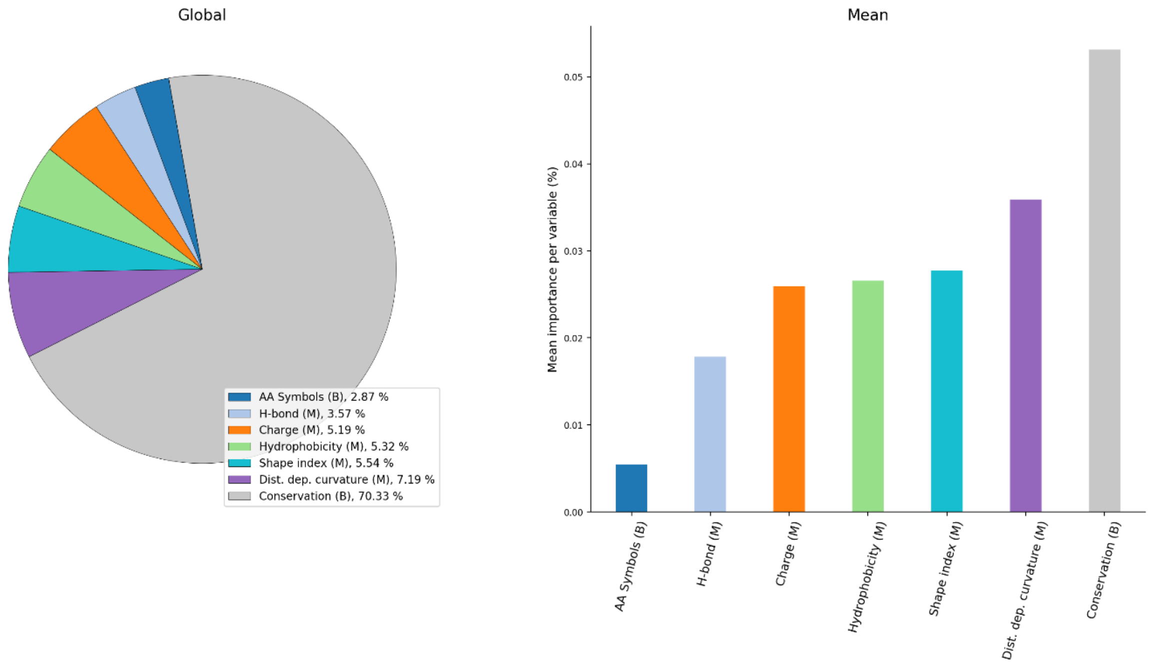

Figure 17: Global and mean feature importances associated with the features used in the training of *BIPSPI-patch-sorted*

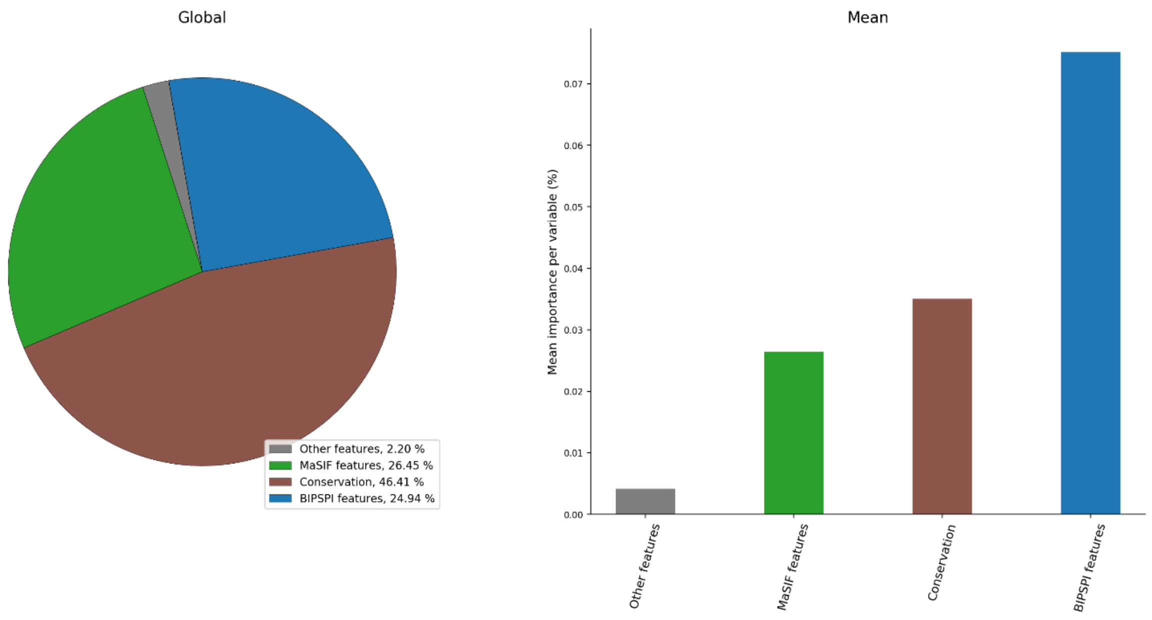

Figure 18: Global and mean feature importances associated with the features used in the training of *BIPSPI-default-patch*. The structural features associated with BIPSPI and MaSIF are grouped into two separate categories called 'BIPSPI features' and ‘MaSIF features’.

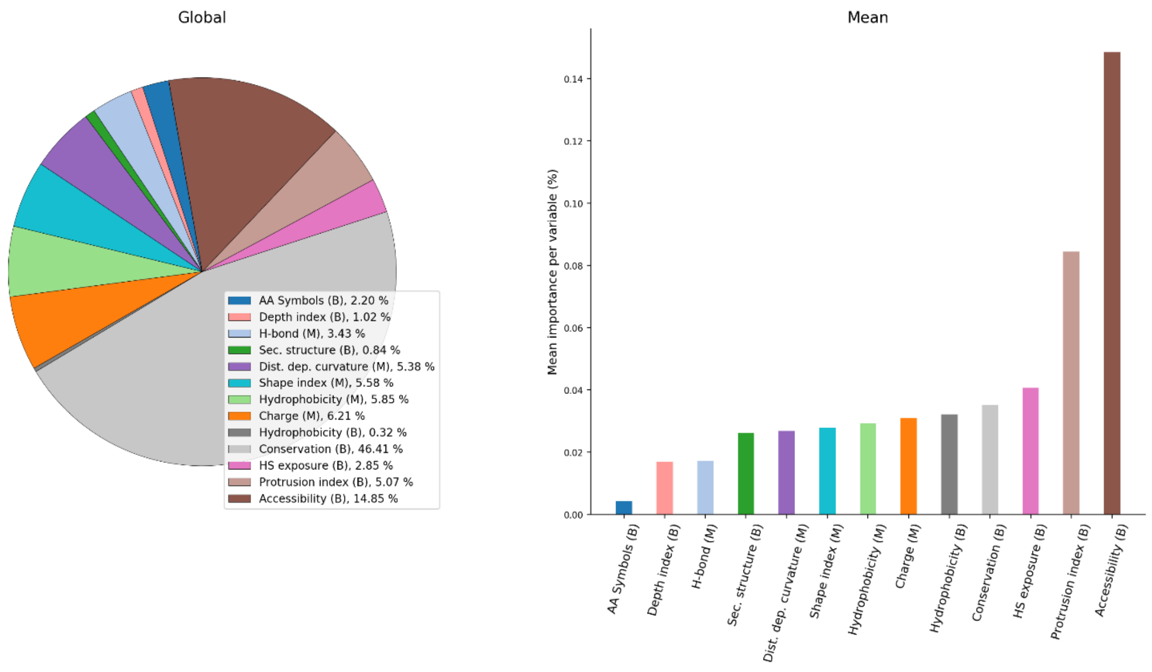

Figure 19: Global and mean feature importances associated with the features used in the training of *BIPSPI-default-patch*

Figures 18 and 19 describe the relative importance of the different features used in the model that combines the structure-derived features from both MaSIF and

BIPSPI. The model was trained with unsorted versions of the patches generated by MaSIF. Similar to the case where the model was trained solely on sequence-based features and unsorted MaSIF patches (*BIPSPI-patch*), the impact of conservation-based features is significantly lower, accounting for only 46.41% of the total importance. In spite of the fact that there are fewer BIPSPI structural features, both feature sets are approximately equally important to the model for prediction performance: only 166 structural feature variables are associated with BIPSPI, compared to the 500 feature variables associated with MaSIF per residue. It is interesting to note that while the hydrophobicity features computed by MaSIF have a significantly larger global impact than those computed by BIPSPI, the two are highly comparable from the perspective of mean importance. Accessibility continues to be the feature with the highest mean importance per variable, with amino acid identities being the least important.

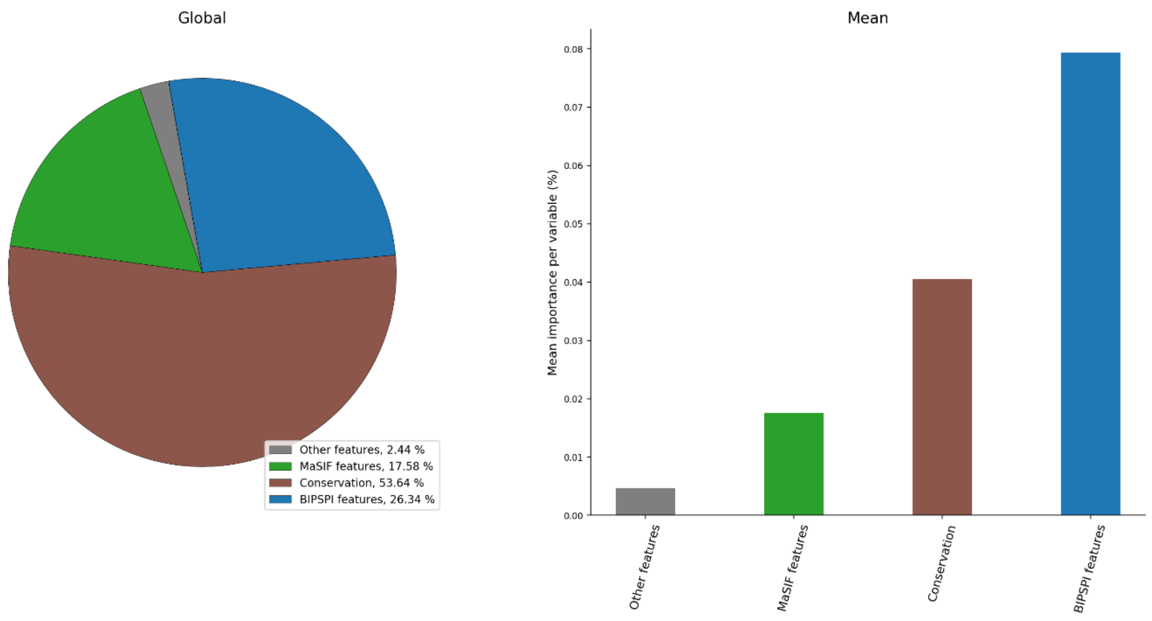

Figure 20: Global and mean feature importances associated with the features used in the training of *BIPSPI-default-patch-sorted*. The structural features associated with BIPSPI and MaSIF are grouped into two separate categories called 'BIPSPI features' and ‘MaSIF features’.

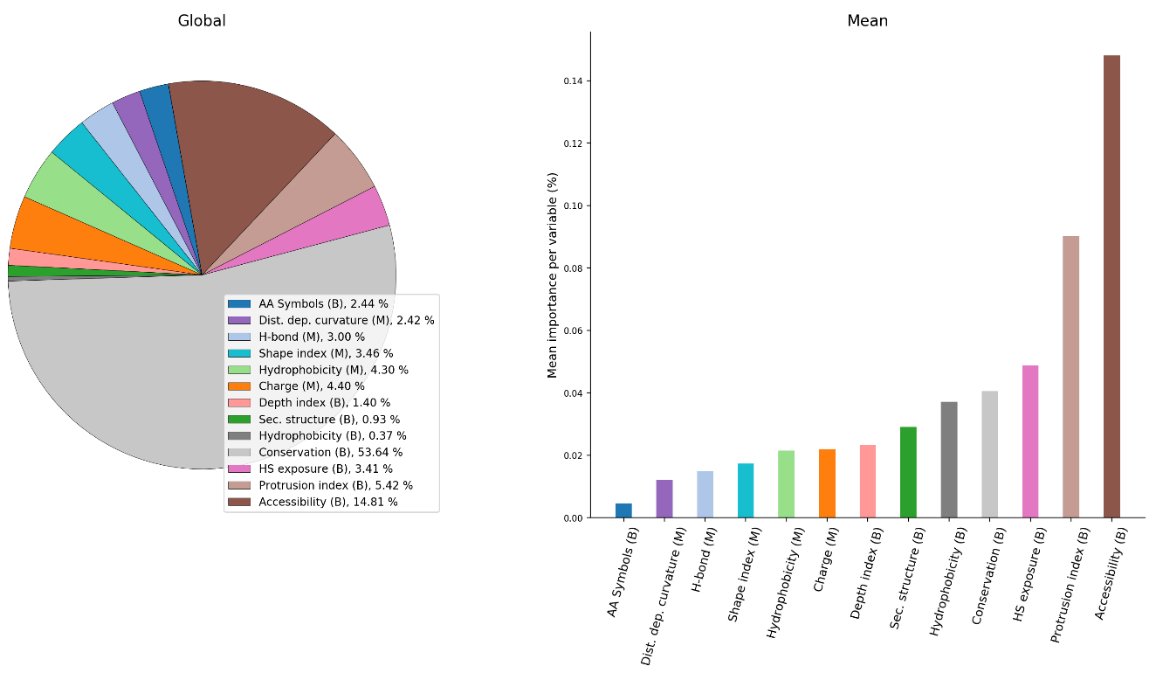

Figure 21: Global and mean feature importances associated with the features used in the training of *BIPSPI-default-patch-sorted*

Figures 20 and 21 illustrate the feature importances computed for a 1-step XGBoost classifier trained on structural features extracted from BIPSPI and MaSIF. The MaSIF patches used to train the model are sorted. We observe that the model is more strongly impacted by BIPSPI-derived features than MaSIF-derived features. BIPSPI features retain a similar fraction of global impact here as they do in *BIPSPI-default-patch*, with the relative proportions amongst the BIPSPI features being conserved as well. Out of the BIPSPI-derived structural features, accessibility is the most important, accounting for 14.81% of the total importance. From the set of features contained in the MaSIF patches, charge is the most impactful, making up 4.40% of the total importance. Sorting patches, in a close parallel to Figure 16, appears to reduce their influence in making predictions. The geometrical features computed by MaSIF appear to be less important to the performance of this model than the chemical features. *BIPSPI-default-patch-sorted* places less importance on the conservation class of features than *BIPSPI-default*. Such features account for 53.64% of the total importance in *BIPSPI-default-patch-sorted*, compared to the 64.98% importance ascribed to them in *BIPSPI-default*. We observe from the mean importance plots for this model that the BIPSPI features appear to be more efficient in relaying information compared to those from MaSIF, evidenced by their higher mean importances per variable.

3.1.2 Discussion

Our objective was to ascertain if the addition of geometrical and chemical features derived from the surfaces of proteins in the form of patches could improve the performance of BIPSPI, a partner-specific interface predictor. Through our experiments, we have demonstrated that protein surface patches are informative features for the purpose of residue-residue contact prediction at protein-protein interfaces. The addition of MaSIF-derived features to the sequence-only model improves ROC-AUC mean performance from 0.8222 to 0.9080, confirming the relevance of the newly added features.

We observe that the addition of MaSIF-derived patches to BIPSPI's existing feature set has a positive effect on the method's performance on the task of residue-residue contact prediction. Both models that combine MaSIF-generated patches and structural features derived from BIPSPI, *BIPSPI-default-patch-sorted* and *BIPSPI-default-patch*, exhibit performance superior to that of *BIPSPI-default* across the most relevant tested metrics. While the addition of patches has a positive impact on model performance, it is interesting to note that the internal sorting of patches improves performance marginally further.

Out of the five features added via the use of MaSIF-generated patches, distance-dependent curvature is consistently the most important across all models trained with them. Protein-protein binding is strongly influenced by geometrical and chemical complementarity. Distance-dependent curvature measures the curvature around each vertex as a function of distance - it is likely that the models have learnt which combinations of the vertex-level distance-dependent curvatures across ligand and receptor residues correspond to potentially complementary surfaces.

From BIPSPI's structure-derived feature set, accessibility appears to have the greatest importance from the perspective of both global and mean importance per variable. This aligns well with literature attesting to the capacity of Solvent Accessible Surface Area to be used for protein-protein and protein-nucleic acid interface hotspot prediction (Martins *et al.*, 2014; Munteanu *et al.*, 2015). Relative accessible surface area has also been reported to be an effective predictor of the extent of binding-induced conformational change in protein complexes (Marsh and Teichmann, 2011), potentially hinting at its utility in predicting protein-protein interfaces. Protrusion index is another structural feature computed by BIPSPI that has a considerable impact on the accuracy of the models tested. Protrusion has been shown in the literature to correlate to a certain extent with accessibility and, in conjunction with other features, has been used for the allied problem of protein-protein interface hot-spot prediction. It is surprising to note that protrusion has a significantly higher impact on performance than shape index, a property that is expected to capture similar information by encapsulating the local curvature around a residue.

It appears that most of BIPSPI's existing structural features are more efficient at relaying structural information than MaSIF's. In our best-performing model, BIPSPI's features were of greater importance than MaSIF's at a global level (26.34% vs 17.58%) and were far more impactful from the perspective of mean importance per variable. From our analysis of feature importances for all five models using structural features, we observe that the use of unsorted patches significantly changes the proportion of impact associated with the various features during the training process. Generally, the use of unsorted patches appears to increase the impact of MaSIF-based features on the models' predictions. Sorting patches internally with respect to their five constituent features reduces these proportions significantly while improving performance. When models using sorted and unsorted patches are analyzed from the perspective of the number of splits each feature is involved in, it is observed that the relative number of splits involving MaSIF-derived features is significantly higher in the latter. Figure 22 illustrates the number of splits involving MaSIF-derived features per tree against the iteration index of the XGBoost models for *BIPSPI-default-patch-sorted* and *BIPSPI-default-patch*. We observe that in the early stages of training, the model using unsorted patches generates trees that contain a much higher number of MaSIF-associated splits than the model using sorted patches.

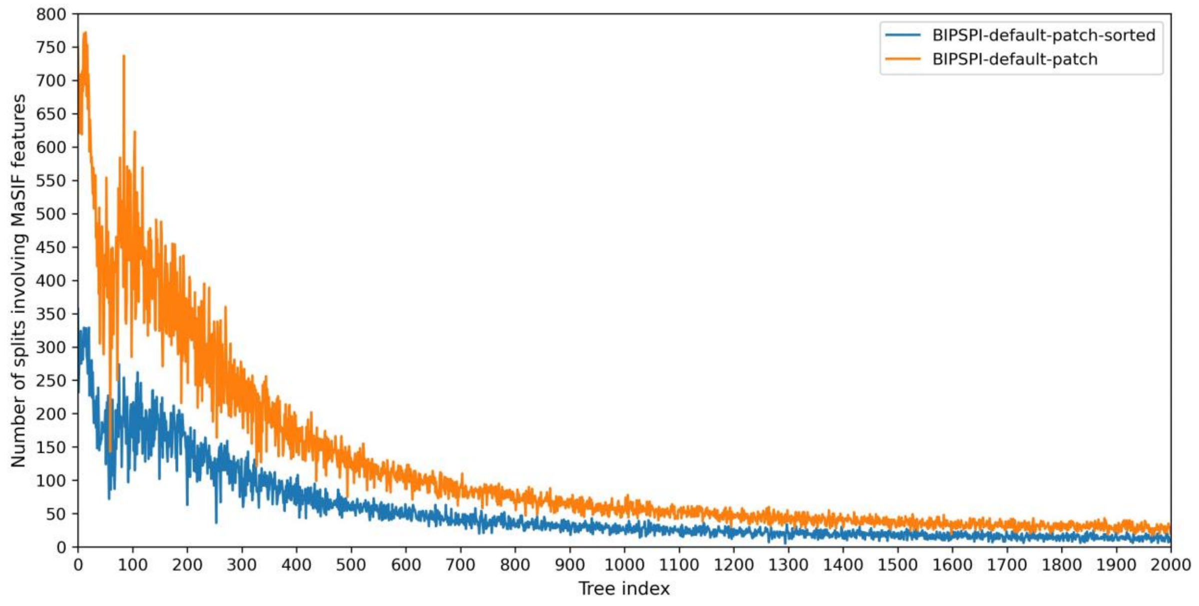

Figure 22: Comparison of the number of splits involving MaSIF features as a function of tree index for BIPSPI-default-patch-sorted and BIPSPI-default-patch

We suspect that this behaviour is connected to the significantly higher variance associated with each index of the patch representation over all the patches the model is exposed to. While sorting a patch compromises the inherent geometry associated with it, it imposes the same structure on all patches in the dataset. The imposition of order greatly reduces the variance in values observed across patches at a particular index. Better put, unlike the case of unsorted patches, each position in the numerical representation of a sorted patch has the *same meaning.* For instance, in a sorted patch consisting of 100 vertices described by the five features computed by MaSIF, the first row will always correspond to the magnitude of the charge on the most negatively charged vertex, the shape index of the vertex with the highest concavity, the vertex with the highest hydrogen bond acceptor potential and similarly so. The next row will correspond to the sequentially subsequent values for the abovementioned properties. We believe that the improvement in performance upon sorting patches arises as a result of this imposition of meaning on the indices of the linearized representation of the patch.

The conservation-based class of features, in most cases, has the highest global impact on the model's performance. The extent to which conservation influences predictions is unsurprising - residues that constitute the interfaces of protein-protein interactions are likely to have been preserved over the course of evolution, as protein-protein interactions mediate several essential life processes (Choi *et al.*, 2009; Teppa *et al.*, 2017). We believe it could be fruitful to extend the notion of evolutionary conservation (and co-evolution) to the level of the protein surface patch to provide learning methods with a notion of the extent of evolutionary complementarity exhibited by a pair of patches. We expect methods trained with patch-level evolutionary information to exhibit superior performance to those trained solely with residue-level conservation data and hope to construct an appropriate representation for this purpose in the future. Furthermore, in addition to conventional sequence-derived features, we also believe that the use of context-aware residue representations produced using protein language models such as ESM-1b (Rives *et al.*, 2021) and ProtT5 (Elnaggar *et al.*, 2022) has the potential to improve performance on this prediction task.

While the use of protein surface patches has improved the performance of BIPSPI, we cognize that a gradient-boosting-based framework is unlikely the best architecture to utilize them to their true potential. XGBoost and other machine-learning frameworks that accept only linear forms of input are inherently limited in their ability to utilize geometric information. A patch, fundamentally, is an area on the surface of the protein: while properties such as shape index and distance-dependent curvature capture the local geometry around a vertex, the positional relationships between various vertices within a given patch are lost during the process of linearization. It is plausible that a convolutional neural network-based architecture or a graph neural network that operates directly on the graph induced by the surface mesh of the protein is likely to exhibit better performance on the task of partner-specific protein interface prediction than a framework such as XGBoost that requires linearization. An alternate approach would be to randomly sample various rotations of a given patch and create multiple representations of the same residue-pair, each using linearizations of different rotations of their constituent patches. The model could subsequently be trained with these representations – this would show the learning method that multiple rotations of a given patch correspond to the same information.

3.2 Data-driven compression methods for protein surface patches

3.2.1 Results

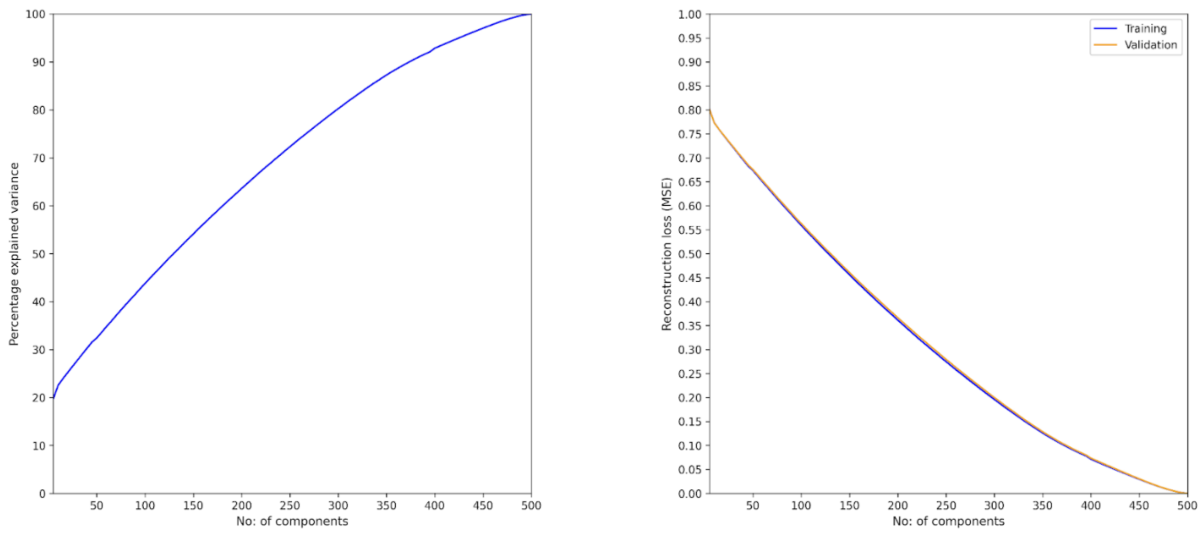

Figure 23: Percentage explained variance and reconstruction loss (MSE) as a function of the number of principal components used for the principal component analysis method fitted to the dataset of unsorted patches

We observe suboptimal compression performance in the case where principal component analysis is performed on a dataset constituted of unsorted patches. Figure 23 illustrates the percentage variance explained as a function of the number of principal components considered and the loss incurred when the input is reconstructed from the first n principal components. To obtain a reconstruction loss of less than 0.05, over 425 principal components are required, indicating that the capacity of unsorted patches to be compressed by principal component analysis is minimal. We also observe that the reconstruction loss computed on the validation set closely mirrors that computed on the training set.

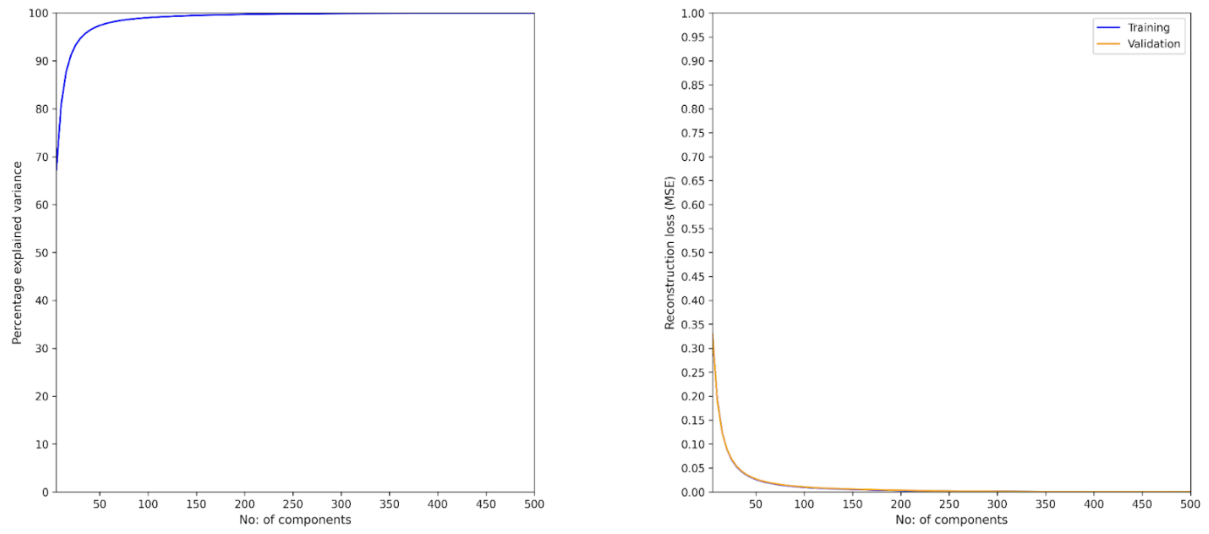

Figure 24: Percentage explained variance and reconstruction loss (MSE) as a function of the number of principal components used for principal component analysis fitted to the dataset of sorted patches

In stark contrast to the case of unsorted patches, principal component analysis appears to perform exceptionally well in the case where the method is fit to sorted patches (Figure 24). The method achieves a reconstruction loss of less than 0.05 with less than 35 components, and crossing a reconstruction loss of 0.01 requires only 105 components. Similar to what was observed in the case of unsorted patches, the reconstruction losses incurred on the training and validation sets mirror each other very closely. There appears to be an inflection point in both plots at approximately 30 components, from where performance begins to asymptote.

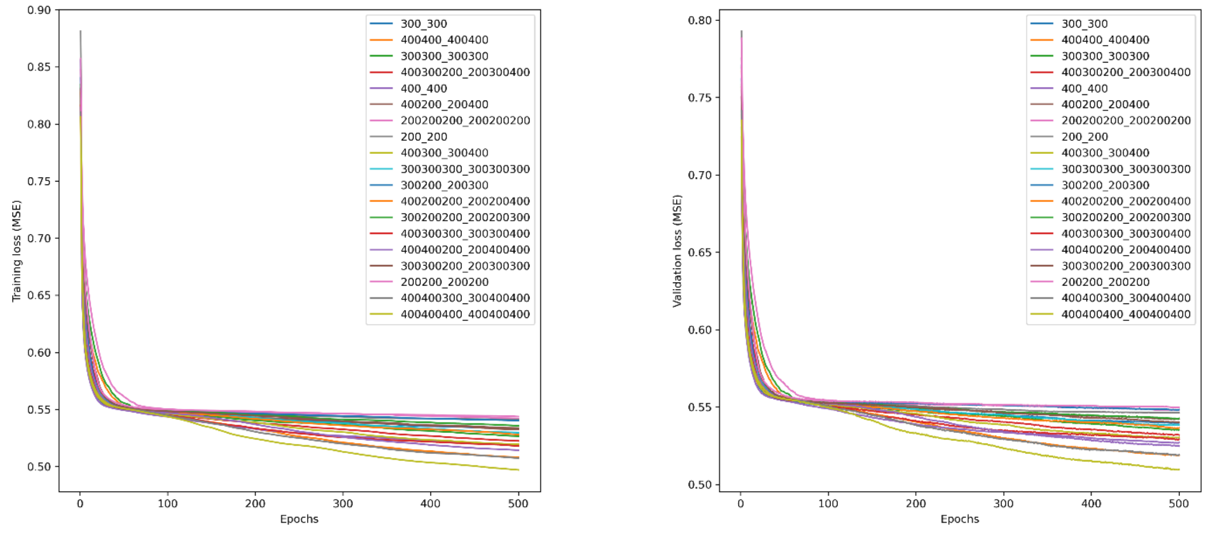

Figure 25: Training and validation loss (MSE) vs training epoch index for autoencoder models with latent dimension size of 100 trained on unsorted patches

We subsequently test the performance of the autoencoder models trained with unsorted patches. The average reconstruction loss on the validation set across all 19 tested models is 0.5352. The best-performing model (MSE = 0.5108) was the most complex, consisting of three hidden layers (each of 400 hidden units) each for the encoder and decoder. The training and validation losses for the 19 models contained in Table 3 are computed by averaging results over the final 50 epochs of training to account for noise. We conclude that the autoencoder architectures tested are unsuitable for the lossless compression of unsorted patches generated by MaSIF. From Table 3 and Figure 25, we do not observe a clear relationship between model complexity and validation set performance.

Table 3: Training and validation losses (MSE) averaged over the final 50 iterations of training for all 19 tested models with latent dimension size of 100 trained with unsorted patches

Architecture	**Training Loss**	**Validation Loss**
400400400_400400400	0.49862033	0.51084647
400400_400400	0.50906376	0.51960411
400400300_300400400	0.5086854	0.52003612
400400200_200400400	0.51521921	0.52563242
400_400	0.51966519	0.52751778
400300300_300300400	0.5190646	0.52996238
400300_300400	0.52020196	0.53064663
400300200_200300400	0.52332491	0.53270997
300300_300300	0.52759914	0.53614796
400200200_200200400	0.52944333	0.53757512
300300300_300300300	0.53029686	0.53863717
300_300	0.5338754	0.54014334
300300200_200300300	0.53314305	0.54084669
400200_200400	0.53431434	0.54268245
300200200_200200300	0.53652639	0.54318872
200_200	0.54138824	0.54646923
200200200_200200200	0.54265775	0.54829711
300200_200300	0.54070021	0.54859973
200200_200200	0.54426069	0.55018174

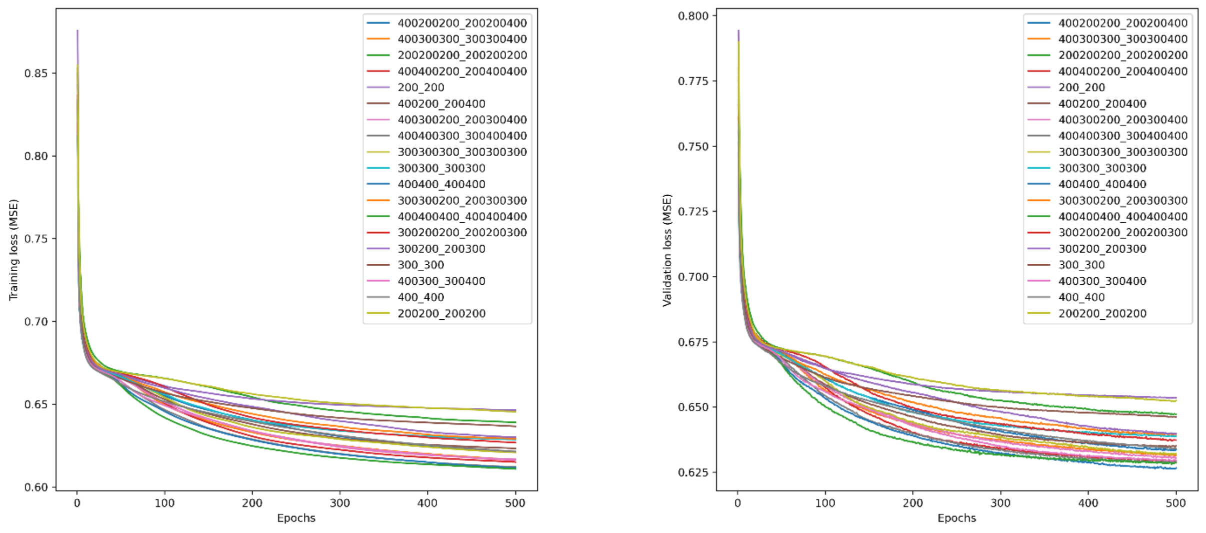

Figure 26: Training and validation losses (MSE) vs training epoch index for autoencoder models with latent dimension size of 50 trained on unsorted patches.

We observe from Figure 26 that performance is significantly worse when the size of the latent dimension is halved from 100 to 50. The validation loss averaged across all 19 models increases from MSE 0.5352 to MSE 0.6359. The models employing 400-dimensional hidden layers appear to exhibit better performance than those using only 300 and 200-dimensional hidden layers. The decrease in performance compared to the models using a bottleneck size of 100 dimensions is expected as the reduction in latent dimension size corresponds to a significantly harder compression task. Table 4 lists the training and validation losses for all 19 models using unsorted patches in conjunction with a latent dimension size of 50 hidden units.

Table 4: Training and validation losses (MSE) averaged over the final 50 iterations of training for all 19 tested models with latent dimension size of 50 trained with unsorted patches.

Architecture	Training Loss	Validation Loss
400400_400400	0.61223592	0.62675255
400400400_400400400	0.61156837	0.62865261
400400300_300400400	0.61261756	0.62898205
400400200_200400400	0.61570714	0.6295598
400300_300400	0.61722149	0.62983679
400300200_200300400	0.61689694	0.63103127
400300300_300300400	0.61724137	0.63197572

300300300_300300300	0.62145022	0.63232813
400200200_200200400	0.62203821	0.63382337
400_400	0.62227445	0.63467881
400200_200400	0.62378838	0.63509277
300200200_200200300	0.62751627	0.6377372
300300_300300	0.6290283	0.63906111
300300200_200300300	0.62955692	0.63977616
300200_200300	0.6306161	0.64027539
300_300	0.63724325	0.64659616
200200200_200200200	0.63955658	0.64766513
200200_200200	0.64607298	0.65275298
200_200	0.64667923	0.65371706

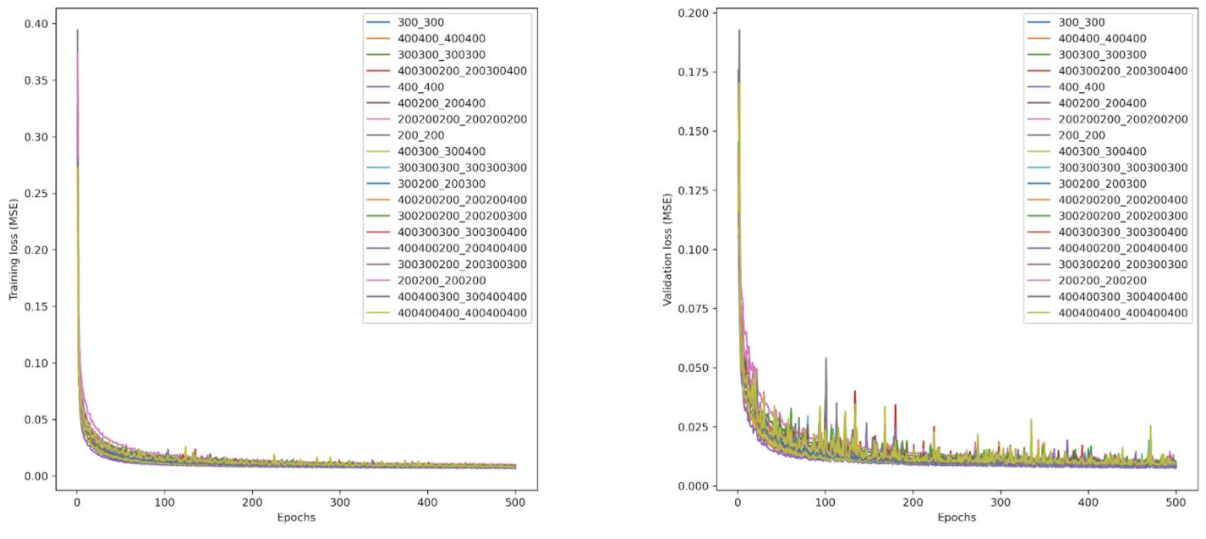

Figure 27: Training and validation losses (MSE) vs training epoch index for autoencoder models with latent dimension size of 100 trained on sorted patches.

From Figure 27, we observe that the autoencoder models trained on sorted patches exhibit vastly superior performance to identical models trained on unsorted patches. Table 5 contains the mean training and validation losses achieved for all 19 models over the final 50 epochs of model training.

Table 5: Training and validation losses (MSE) averaged over the final 50 iterations of training for all 19 tested models with latent dimension size of 100 trained with sorted patches

Architecture	Training Loss	Validation Loss
400_400	0.00692446	0.00782732
300_300	0.00717828	0.00809767
300300_300300	0.00748921	0.00867917
200_200	0.00785857	0.00885369
400300_300400	0.00769637	0.00885858
300200_200300	0.00785881	0.00897441
400200_200400	0.0078138	0.00911418
400400_400400	0.00782203	0.00920799
200200_200200	0.00839622	0.00952031
400400200_200400400	0.00820127	0.00962707
400300200_200300400	0.0085139	0.00975329
400300300_300300400	0.00855657	0.00982589
400200200_200200400	0.00846929	0.00991788
400400300_300400400	0.00849327	0.0099296
300300200_200300300	0.00873274	0.00995838
300300300_300300300	0.00863568	0.01010596
400400400_400400400	0.00860751	0.01024261
300200200_200200300	0.00940295	0.01075625
200200200_200200200	0.01013257	0.01128386

The best model achieves an average reconstruction loss of 0.0078 on the validation dataset over the final 50 epochs of training. When trained for 500 epochs, it achieves a test set reconstruction loss of 0.0073. This is highly competitive and allows for compression with exceptionally minimal information loss. The performances of the models were compared using the mean of the validation losses computed for the last 50 epochs of training. While all models are significantly more competitive than their counterparts trained on unsorted patches, it is interesting to note that increasing model size does not appear to result in substantial improvements to validation set performance. In fact, we observe that the simpler models exhibit better performance than models that have a greater degree of complexity – models with single-layer encoders and decoders perform better than those with two and three layers. The models exhibit a mean validation set MSE of 0.0095.

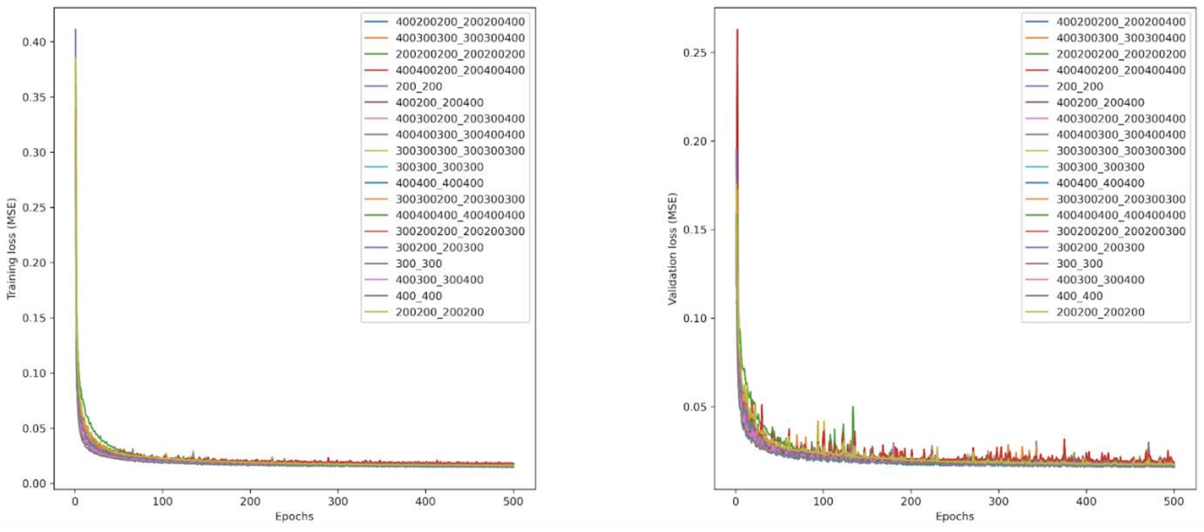

Figure 28: Training and validation losses (MSE) vs training epoch index for autoencoder models with latent dimension size of 50 trained on sorted patches.

Encouraged by the exceptional performance of the models with latent dimension size 100, we tested the same architectures with a latent dimension size of 50 (Figure 28). This increased the extent of compression from 5x to 10x. Table 6 contains the mean training and validation losses achieved for all 19 models over the final 50 epochs of model training.

Table 6: Training and validation losses (MSE) averaged over the final 50 iterations of training for all 19 tested models with latent dimension size of 50 trained with sorted patches.

Architecture	Training Loss	Validation Loss
400400_400400	0.014832	0.01628916
400_400	0.01525211	0.01638039
400300_300400	0.01526113	0.01658419
300300_300300	0.0154943	0.01674393
400200_200400	0.01558173	0.01690069
300_300	0.01577537	0.01697213
400400400_400400400	0.01537724	0.01697219
400300300_300300400	0.0156564	0.01706572
400200200_200200400	0.0160742	0.01745806

200200_200200	0.01637748	0.01764264
400400200_200400400	0.01611265	0.01768556
300300300_300300300	0.01616417	0.01769364
300200_200300	0.0163793	0.01769536
400400300_300400400	0.01610283	0.0179201
400300200_200300400	0.01628153	0.01804864
300300200_200300300	0.01674769	0.0181483
200200200_200200200	0.01713168	0.01850336
200_200	0.01756568	0.01872389
300200200_200200300	0.01828892	0.01982203

The best model achieves an average reconstruction loss of 0.0163 on the validation dataset over the final 50 epochs of training. When trained for 500 epochs, it achieves a test set reconstruction loss of 0.0159. On average, the models perform marginally worse than those with a latent dimension size of 100. The mean validation loss exhibited across the tested models is 0.0175. The trend of less complex models exhibiting marginally superior performance is visible with this class of models as well.

3.2.2 Discussion

Across both methods of compression, we observe that performance improves by several orders of magnitude when the patches in the training and validation datasets are sorted prior to training. For both sorted and unsorted patches, the autoencoder models built with 100-dimensional bottleneck layers exhibited better performance than those trained with 50-dimensional bottlenecks. This is expected as the latter represents a considerably higher degree of compression. As is expected for principal component analysis, increasing the number of principal components used in the reconstruction decreases the magnitude of loss incurred.

In the context of the training and validation sets used, the sorted variant of the dataset exhibits considerably lower variances for 491 of the 500 variables across patches in comparison to the variant of the dataset where patches are unsorted. Similar to the case observed during the training of BIPSPI with sorted patches, sorting patches specifies the interpretation associated with each index of the patch vector. This is likely the reason why it appears to be significantly easier for learning-based compression methods to compress a sorted patch than an unsorted patch. We reiterate that sorting happens *within* a patch along the five MaSIF features and not at the level of the dataset.

From our experiments, we conclude that sorting is a necessary prerequisite for the successful compression of protein surface patches. The methods developed and tested by us are highly competent at the task of compressing sorted protein surface patches. We believe that the compressed representations can be directly used to train protein-protein interface prediction models such as BIPSPI that are reliant on linear input representations or for other allied problems such as protein-ligand binding prediction or interface hotspot prediction. The significantly smaller size of the patch representation can either facilitate faster training owing to the reduced size of the input data or make space to allow for the use of additional sequence/structure-derived features, such as context-aware residue representations created by protein language models.

To compare the two unsupervised learning methods, the PCA-based methods using 50 and 100 principal components were tested on the held-out test set. At a compressed representation size of 100 dimensions, the PCA-based method exhibits marginally higher reconstruction loss (MSE 0.0102) on the test set than that incurred by the most performant comparable autoencoder model (MSE 0.0073). The difference is significantly higher at a compressed representation size of 50 dimensions (MSE 0.0269 vs MSE 0.0159). While the autoencoder appears to be more performant at the task of compression, we believe that the choice of method should depend on the user's priorities. If minimizing reconstruction error to near-lossless levels is the objective, the autoencoder method is preferred. However, in the event that interpretability and ease of use are of greater importance, the PCA method is a sufficiently competent alternative.

As sorting corrupts the geometrical relationships between the various vertices that constitute a patch, we acknowledge that the linearized compressed patch representation is likely to be suboptimal when used in conjunction with learning algorithms that can take positional information regarding the various inputs into account. To create compressed input representations for such models, we recommend using rotation-invariant deep-learning-based models. We believe the polar convolutional neural network-based method MaSIF uses to generate fingerprints can be adapted for the purpose of patch compression, owing to its ability to learn local geometrical patterns within patches using convolutional filters and its integration of multiple rotations of each patch to incorporate rotation invariance.

References

Ahmad, S, and Mizuguchi, K (2011). Partner-Aware Prediction of Interacting Residues in Protein-Protein Complexes from Sequence Data. PLOS ONE 6, e29104.

Alberstein, RG, Guo, AB, and Kortemme, T (2022). Design principles of protein switches. Curr Opin Struct Biol 72, 71–78.

Altschul, S (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research 25, 3389–3402.

Anderson, MR, and Cafarella, M (2016). Input selection for fast feature engineering. In: 2016 IEEE 32nd International Conference on Data Engineering (ICDE), 577–588.

Baek, M et al. (2021). Accurate prediction of protein structures and interactions using a three-track neural network. Science 373, 871–876.

Bahadur, RP, and Zacharias, M (2008). The interface of protein-protein complexes: Analysis of contacts and prediction of interactions. Cell Mol Life Sci 65, 1059–1072.

Bai, X, McMullan, G, and Scheres, SHW (2015). How cryo-EM is revolutionizing structural biology. Trends in Biochemical Sciences 40, 49–57.

Bank, D, Koenigstein, N, and Giryes, R (2021). Autoencoders.

Benjin, X, and Ling, L (2020). Developments, applications, and prospects of cryo-electron microscopy. Protein Sci 29, 872–882.

Cao, L et al. (2022). Design of protein-binding proteins from the target structure alone. Nature 605, 551–560.

Chen, R, Mintseris, J, Janin, J, and Weng, Z (2003). A protein-protein docking benchmark. Proteins 52, 88–91.

Chen, T, and Guestrin, C (2016). XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794.

Chen, X et al. (2018). DCZ3112, a novel Hsp90 inhibitor, exerts potent antitumor activity against HER2-positive breast cancer through disruption of Hsp90-Cdc37 interaction. Cancer Lett 434, 70–80.

Choi, YS, Yang, J-S, Choi, Y, Ryu, SH, and Kim, S (2009). Evolutionary conservation in multiple faces of protein interaction. Proteins 77, 14–25.

Clarke, M (2010). Muscle sliding filaments. Nat Rev Mol Cell Biol 9, s7–s7.

Cock, PJA et al. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics 25, 1422–1423.

Cong, Q, Anishchenko, I, Ovchinnikov, S, and Baker, D (2019). Protein interaction networks revealed by proteome coevolution. Science 365, 185–189.

Dai, B, and Bailey-Kellogg, C (2021). Protein interaction interface region prediction by geometric deep learning. Bioinform.

Dall'Acqua, W et al. (1998). A Mutational Analysis of Binding Interactions in an Antigen−Antibody Protein−Protein Complex. Biochemistry 37, 7981–7991.

Eddy, SR (1995). Multiple alignment using hidden Markov models. Proc Int Conf Intell Syst Mol Biol 3, 114–120.

Elnaggar, A et al. (2022). ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning. IEEE Trans Pattern Anal Mach Intell 44, 7112–7127.

Evans, R et al. (2022). Protein complex prediction with AlphaFold-Multimer. 2021.10.04.463034.

Finn, RD, Clements, J, and Eddy, SR (2011). HMMER web server: interactive sequence similarity searching. Nucleic Acids Res 39, W29–W37.

Fletcher, DA, and Mullins, RD (2010). Cell mechanics and the cytoskeleton. Nature 463, 485–492.

Fout, A, Byrd, J, Shariat, B, and Ben-Hur, A (2017). Protein Interface Prediction using Graph Convolutional Networks. In: Advances in Neural Information Processing Systems, Curran Associates, Inc.

Friedman, JH (2001). Greedy function approximation: A gradient boosting machine. The Annals of Statistics 29, 1189–1232.

Fuchs, E, and Cleveland, DW (1998). A Structural Scaffolding of Intermediate Filaments in Health and Disease. Science 279, 514–519.

Gainza, P, Sverrisson, F, Monti, F, Rodolà, E, Boscaini, D, Bronstein, MM, and Correia, BE (2020). Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. Nat Methods 17, 184–192.

Ghusinga, KR, Jones, RD, Jones, AM, and Elston, TC (2021). Molecular switch architecture determines response properties of signaling pathways. Proceedings of the National Academy of Sciences 118, e2013401118.

Goyal, A, and Bengio, Y (2022). Inductive biases for deep learning of higher-level cognition. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 478, 20210068.

Green, AG, Elhabashy, H, Brock, KP, Maddamsetti, R, Kohlbacher, O, and Marks, DS (2021). Large-scale discovery of protein interactions at residue resolution using co-evolution calculated from genomic sequences. Nat Commun 12, 1396.

Greener, JG, Kandathil, SM, Moffat, L, and Jones, DT (2022). A guide to machine learning for biologists. Nat Rev Mol Cell Biol 23, 40–55.

Guest, JD, Vreven, T, Zhou, J, Moal, I, Jeliazkov, JR, Gray, JJ, Weng, Z, and Pierce, BG (2021). An expanded benchmark for antibody-antigen docking and affinity prediction reveals insights into antibody recognition determinants. Structure 29, 606-621.e5.

Ha, J-H, and Loh, SN (2012). Protein conformational switches: from nature to design. Chemistry 18, 7984–7999.

Hamelryck, T (2005). An amino acid has two sides: a new 2D measure provides a different view of solvent exposure. Proteins 59, 38–48.

Harkey, T, Govind Kumar, V, Hettige, J, Tabari, SH, Immadisetty, K, and Moradi, M (2019). The Role of a Crystallographically Unresolved Cytoplasmic Loop in Stabilizing the Bacterial Membrane Insertase YidC2. Sci Rep 9, 14451.

Herrmann, H, Bär, H, Kreplak, L, Strelkov, SV, and Aebi, U (2007). Intermediate filaments: from cell architecture to nanomechanics. Nat Rev Mol Cell Biol 8, 562–573.

Hopf, TA et al. (2019). The EVcouplings Python framework for coevolutionary sequence analysis. Bioinformatics 35, 1582–1584.

Hopf, TA, Schärfe, CPI, Rodrigues, JPGLM, Green, AG, Kohlbacher, O, Sander, C, Bonvin, AMJJ, and Marks, DS (2014). Sequence co-evolution gives 3D contacts and structures of protein complexes. ELife 3, e03430.

Hou, Z, Yang, Y, Ma, Z, Wong, K, and Li, X (2023). Learning the protein language of proteome-wide protein-protein binding sites via explainable ensemble deep learning. Commun Biol 6, 1–15.

Hu, Y, Cheng, K, He, L, Zhang, X, Jiang, B, Jiang, L, Li, C, Wang, G, Yang, Y, and Liu, M (2021). NMR-Based Methods for Protein Analysis. Anal Chem 93, 1866–1879.

Huxley, AF, and Niedergerke, R (1954). Structural Changes in Muscle During Contraction: Interference Microscopy of Living Muscle Fibres. Nature 173, 971–973.

Hwang, H, Pierce, B, Mintseris, J, Janin, J, and Weng, Z (2008). Protein–protein docking benchmark version 3.0. Proteins: Structure, Function, and Bioinformatics 73, 705–709.

Jaderberg, M, Simonyan, K, Zisserman, A, and kavukcuoglu, koray (2015). Spatial Transformer Networks. In: Advances in Neural Information Processing Systems, Curran Associates, Inc.

Jeong, H, Tombor, B, Albert, R, Oltvai, ZN, and Barabási, A-L (2000). The large-scale organization of metabolic networks. Nature 407, 651–654.

Jordan, MI, and Mitchell, TM (2015). Machine learning: Trends, perspectives, and prospects. Science 349, 255–260.

Jumper, J et al. (2021). Highly accurate protein structure prediction with AlphaFold. Nature 596, 583–589.

Jurrus, E et al. (2018). Improvements to the APBS biomolecular solvation software suite. Protein Science 27, 112–128.

Kabsch, W, and Sander, C (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22, 2577–2637.

Kim, TY, Cha, JS, Kim, H, Choi, Y, Cho, H-S, and Kim, H-S (2021). Computationally-guided design and affinity improvement of a protein binder targeting a specific site on HER2. Computational and Structural Biotechnology Journal 19, 1325–1334.

Kingma, DP, and Ba, J (2015). Adam: A Method for Stochastic Optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, ed. Y Bengio, and Y LeCun.

Kipf, TN, and Welling, M (2017). Semi-Supervised Classification with Graph Convolutional Networks.

Kortemme, T, Morozov, AV, and Baker, D (2003). An orientation-dependent hydrogen bonding potential improves prediction of specificity and structure for proteins and protein-protein complexes. J Mol Biol 326, 1239–1259.

Kramer, MA (1992). Autoassociative neural networks. Computers & Chemical Engineering 16, 313–328.

Krapp, LF, Abriata, LA, Rodriguez, FC, and Peraro, MD (2022). PeSTo: parameter-free geometric deep learning for accurate prediction of protein interacting interfaces. 2022.05.09.491165.

LeCun, Y, Bengio, Y, and Hinton, G (2015). Deep learning. Nature 521, 436–444.

Lee, B, and Richards, FM (1971). The interpretation of protein structures: Estimation of static accessibility. Journal of Molecular Biology 55, 379-IN4.

Leinonen, R, Diez, FG, Binns, D, Fleischmann, W, Lopez, R, and Apweiler, R (2004). UniProt archive. Bioinformatics 20, 3236–3237.

Li, Z et al. (2017). The OncoPPi network of cancer-focused protein–protein interactions to inform biological insights and therapeutic strategies. Nat Commun 8, 14356.

Lo Conte, L, Ailey, B, Hubbard, TJP, Brenner, SE, Murzin, AG, and Chothia, C (2000). SCOP: a Structural Classification of Proteins database. Nucleic Acids Res 28, 257–259.

Manfredi, M, Savojardo, C, Martelli, PL, and Casadio, R (2023). ISPRED-SEQ: Deep Neural Networks and Embeddings for Predicting Interaction Sites in Protein Sequences. Journal of Molecular Biology, 167963.

Marchand, A, Van Hall-Beauvais, AK, and Correia, BE (2022). Computational design of novel protein–protein interactions – An overview on methodological approaches and applications. Current Opinion in Structural Biology 74, 102370.

Marsh, JA, and Teichmann, SA (2011). Relative Solvent Accessible Surface Area Predicts Protein Conformational Changes upon Binding. Structure 19, 859–867.

Martins, JM, Ramos, RM, Pimenta, AC, and Moreira, IS (2014). Solvent-accessible surface area: How well can be applied to hot-spot detection? Proteins: Structure, Function, and Bioinformatics 82, 479–490.

Mihel, J, Šikić, M, Tomić, S, Jeren, B, and Vlahoviček, K (2008). PSAIA – Protein Structure and Interaction Analyzer. BMC Structural Biology 8, 21.

Milburn, MV, Tong, L, deVos, AM, Brünger, A, Yamaizumi, Z, Nishimura, S, and Kim, SH (1990). Molecular switch for signal transduction: structural differences between active and inactive forms of protooncogenic ras proteins. Science 247, 939–945.

Minhas, F ul AA, Geiss, BJ, and Ben-Hur, A (2014). PAIRpred: partner-specific prediction of interacting residues from sequence and structure. Proteins 82, 1142–1155.

Munteanu, CR, Pimenta, AC, Fernandez-Lozano, C, Melo, A, Cordeiro, MNDS, and Moreira, IS (2015). Solvent Accessible Surface Area-Based Hot-Spot Detection Methods for Protein–Protein and Protein–Nucleic Acid Interfaces. J Chem Inf Model 55, 1077–1086.

Murakami, Y, and Mizuguchi, K (2010). Applying the Naïve Bayes classifier with kernel density estimation to the prediction of protein-protein interaction sites. Bioinformatics 26, 1841–1848.

Ovchinnikov, S, Kamisetty, H, and Baker, D (2014). Robust and accurate prediction of residue–residue interactions across protein interfaces using evolutionary information. ELife 3, e02030.

Pawson, T, and Nash, P (2000). Protein–protein interactions define specificity in signal transduction. Genes Dev 14, 1027–1047.

Pearson, K (1901). LIII. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2, 559–572.

Pei, J, and Grishin, NV (2001). AL2CO: calculation of positional conservation in a protein sequence alignment. Bioinformatics 17, 700–712.

Pintar, A, Carugo, O, and Pongor, S (2002). CX, an algorithm that identifies protruding atoms in proteins. Bioinformatics 18, 980–984.

Pintar, A, Carugo, O, and Pongor, S (2003). DPX: for the analysis of the protein core. Bioinformatics 19, 313–314.

Pittala, S, and Bailey-Kellogg, C (2020). Learning context-aware structural representations to predict antigen and antibody binding interfaces. Bioinformatics 36, 3996–4003.

Plaut, E (2018). From Principal Subspaces to Principal Components with Linear Autoencoders.

Qi, CR, Su, H, Mo, K, and Guibas, LJ (2017). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. 652–660.

Rives, A et al. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proceedings of the National Academy of Sciences 118, e2016239118.

Sanchez-Garcia, R, Macias, JR, Sorzano, COS, Carazo, JM, and Segura, J (2022). BIPSPI+: Mining Type-Specific Datasets of Protein Complexes to Improve Protein Binding Site Prediction. Journal of Molecular Biology 434, 167556.

Sanchez-Garcia, R, Sorzano, COS, Carazo, JM, and Segura, J (2019). BIPSPI: a method for the prediction of partner-specific protein–protein interfaces. Bioinformatics 35, 470–477.

Sanner, MF, Olson, AJ, and Spehner, J-C (1996). Reduced surface: An efficient way to compute molecular surfaces. Biopolymers 38, 305–320.

Sarker, IH (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT SCI 2, 160.

Scott, DE, Bayly, AR, Abell, C, and Skidmore, J (2016). Small molecules, big targets: drug discovery faces the protein–protein interaction challenge. Nat Rev Drug Discov 15, 533–550.

Seemayer, S, Gruber, M, and Söding, J (2014). CCMpred—fast and precise prediction of protein residue–residue contacts from correlated mutations. Bioinformatics 30, 3128–3130.

Shrake, A, and Rupley, JA (1973). Environment and exposure to solvent of protein atoms. Lysozyme and insulin. Journal of Molecular Biology 79, 351–371.

Smyth, MS, and Martin, JHJ (2000). x Ray crystallography. Molecular Pathology 53, 8–14.

Søgaard-Andersen, L, and Valentin-Hansen, P (1993). Protein-protein interactions in gene regulation: the cAMP-CRP complex sets the specificity of a second DNA-binding protein, the CytR repressor. Cell 75, 557–566.

Stites, WE (1997). Protein−Protein Interactions: Interface Structure, Binding Thermodynamics, and Mutational Analysis. Chem Rev 97, 1233–1250.

Suzek, BE, Huang, H, McGarvey, P, Mazumder, R, and Wu, CH (2007). UniRef: comprehensive and non-redundant UniProt reference clusters. Bioinformatics 23, 1282–1288.

Tarca, AL, Carey, VJ, Chen, X, Romero, R, and Drăghici, S (2007). Machine Learning and Its Applications to Biology. PLoS Comput Biol 3, e116.

Teppa, E, Zea, DJ, and Marino-Buslje, C (2017). Protein–protein interactions leave evolutionary footprints: High molecular coevolution at the core of interfaces. Protein Sci 26, 2438–2444.

Tien, MZ, Meyer, AG, Sydykova, DK, Spielman, SJ, and Wilke, CO (2013). Maximum Allowed Solvent Accessibilites of Residues in Proteins. PLoS One 8, e80635.

Vaswani, A, Shazeer, N, Parmar, N, Uszkoreit, J, Jones, L, Gomez, AN, Kaiser, L, and Polosukhin, I (2017). Attention Is All You Need.

Veličković, P, Cucurull, G, Casanova, A, Romero, A, Liò, P, and Bengio, Y (2018). Graph Attention Networks.

Virtanen, P et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods 17, 261–272.

Vreven, T et al. (2015). Updates to the Integrated Protein-Protein Interaction Benchmarks: Docking Benchmark Version 5 and Affinity Benchmark Version 2. J Mol Biol 427, 3031–3041.

Wang, G, and Dunbrack, RL, Jr (2003). PISCES: a protein sequence culling server. Bioinformatics 19, 1589–1591.

Word, JM, Lovell, SC, Richardson, JS, and Richardson, DC (1999). Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation11Edited by J. Thornton. Journal of Molecular Biology 285, 1735–1747.

wwPDB consortium (2019). Protein Data Bank: the single global archive for 3D macromolecular structure data. Nucleic Acids Research 47, D520–D528.

Xue, LC, Dobbs, D, Bonvin, AMJJ, and Honavar, V (2015). Computational prediction of protein interfaces: A review of data driven methods. FEBS Lett 589, 3516–3526.

Yin, S, Proctor, EA, Lugovskoy, AA, and Dokholyan, NV (2009). Fast screening of protein surfaces using geometric invariant fingerprints. Proceedings of the National Academy of Sciences 106, 16622–16626.

Zanotti, G, Folli, C, Cendron, L, Alfieri, B, Nishida, SK, Gliubich, F, Pasquato, N, Negro, A, and Berni, R (2008). Structural and mutational analyses of protein–protein interactions between transthyretin and retinol-binding protein. The FEBS Journal 275, 5841–5854.

Zheng, H, Handing, KB, Zimmerman, MD, Shabalin, IG, Almo, SC, and Minor, W (2015). X-ray crystallography over the past decade for novel drug discovery – where are we heading next? Expert Opin Drug Discov 10, 975–989.

(2004). Hypervariable region. In: Rheumatology and Immunology Therapy, ed. JD Abbott et al., Berlin, Heidelberg: Springer, 424–424.

Appendix

Feature importance for 2-step classifiers (Gain).

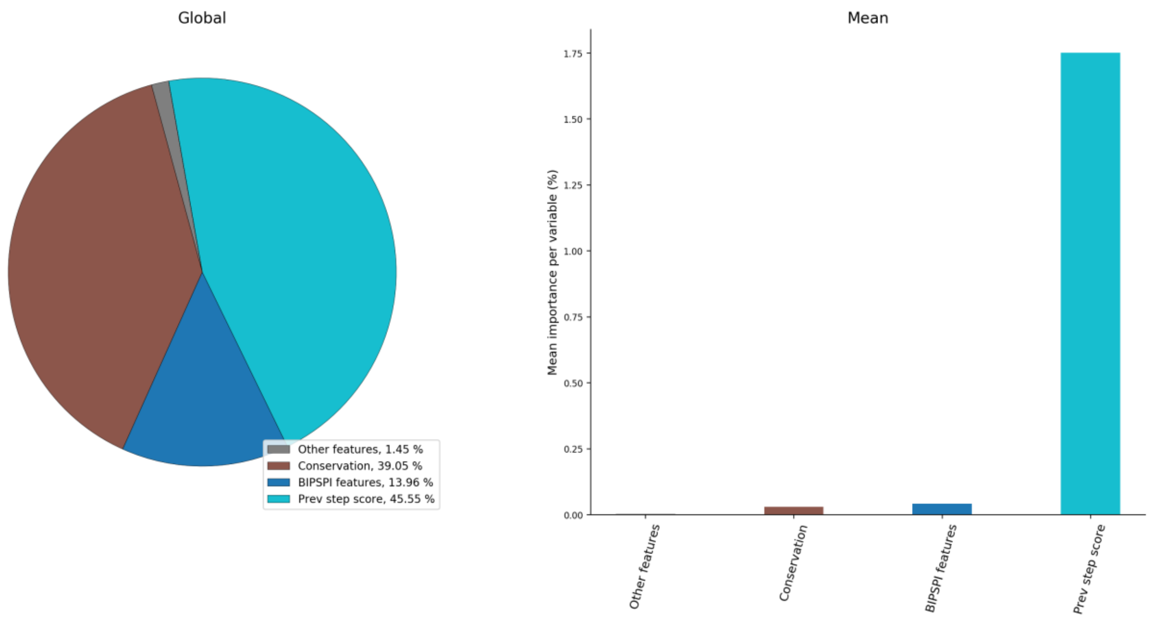

Figure 29: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-default*. The structural features associated with BIPSPI are grouped together in a single category called 'BIPSPI features'.

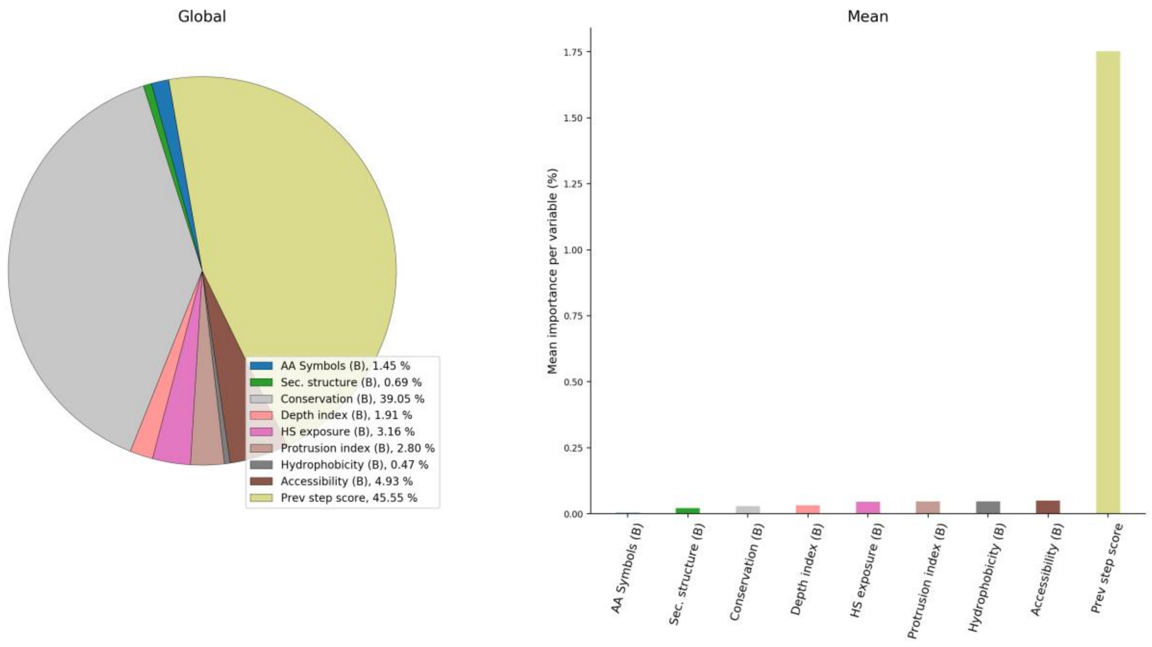

Figure 30: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-default*

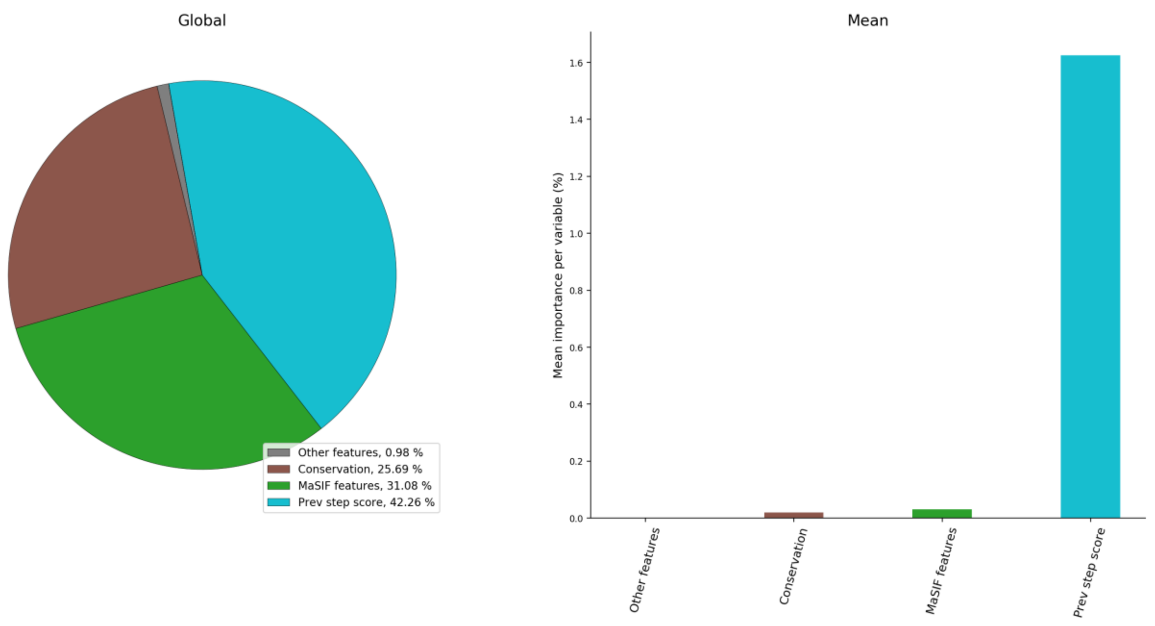

Figure 31: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-patch*. The structural features associated with MaSIF are grouped together in a single category called 'MaSIF features'.

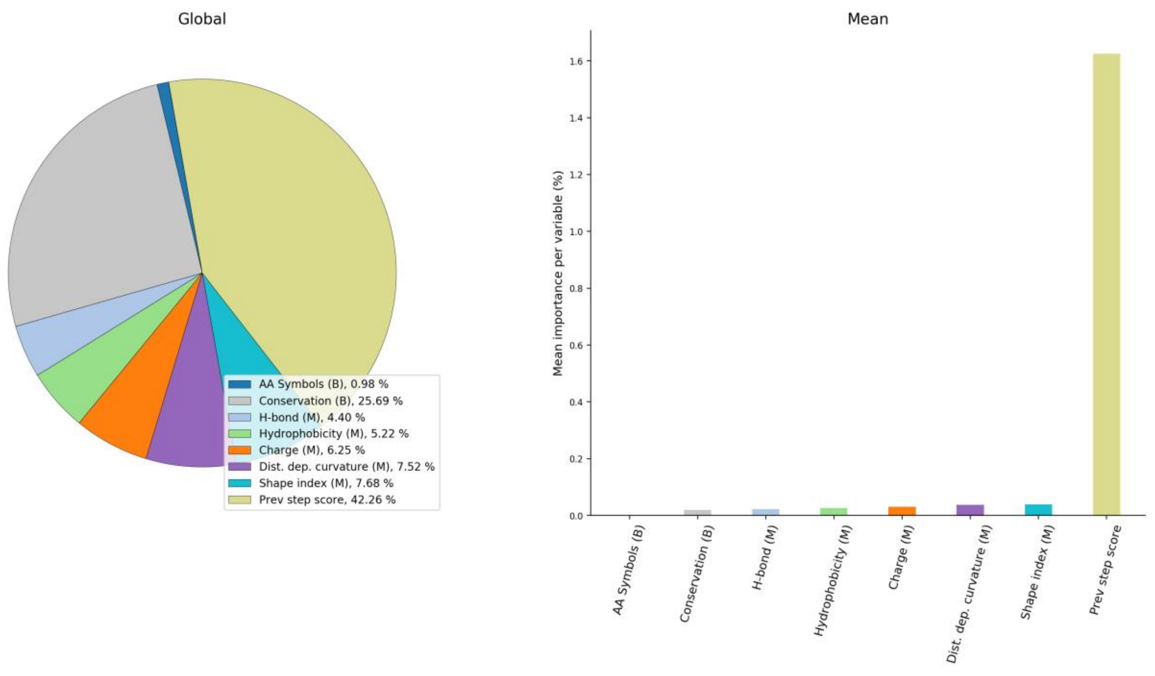

Figure 32: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-patch*

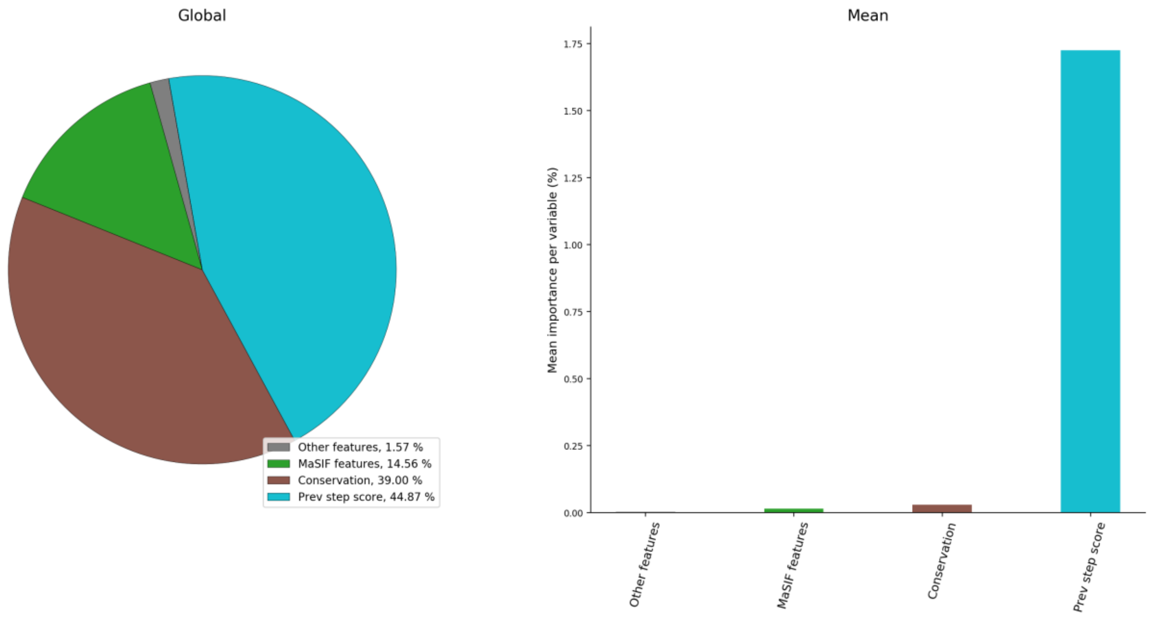

Figure 33: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-patch-sorted*. The structural features associated with MaSIF are grouped together in a single category called ‘MaSIF features’.

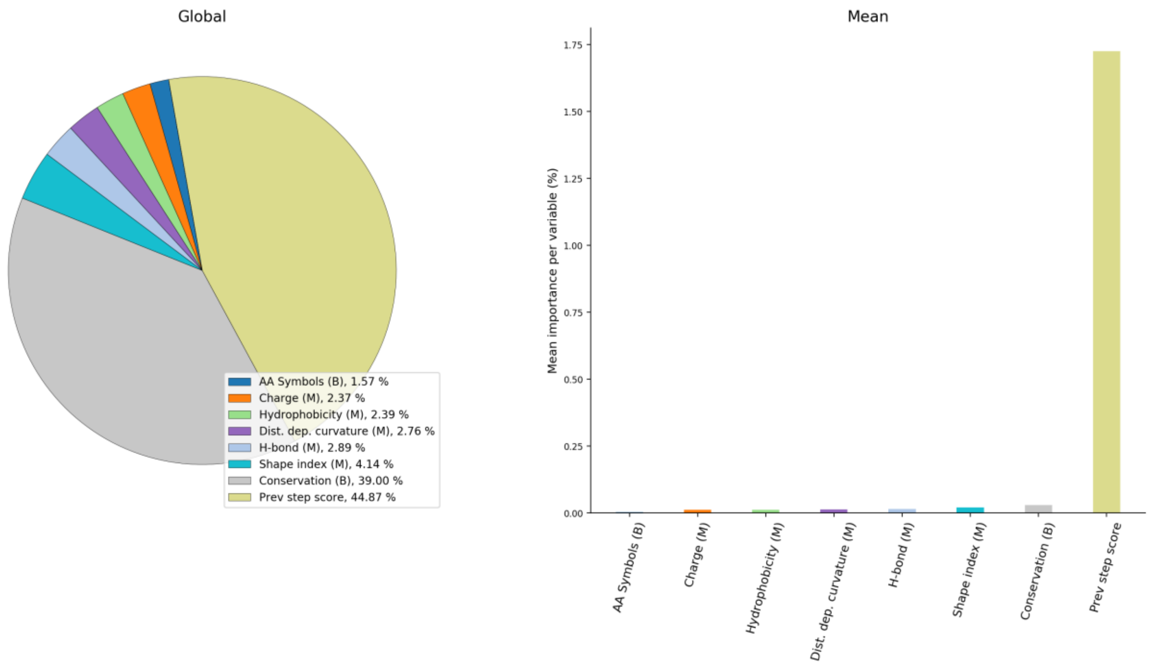

Figure 34: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-patch-sorted*

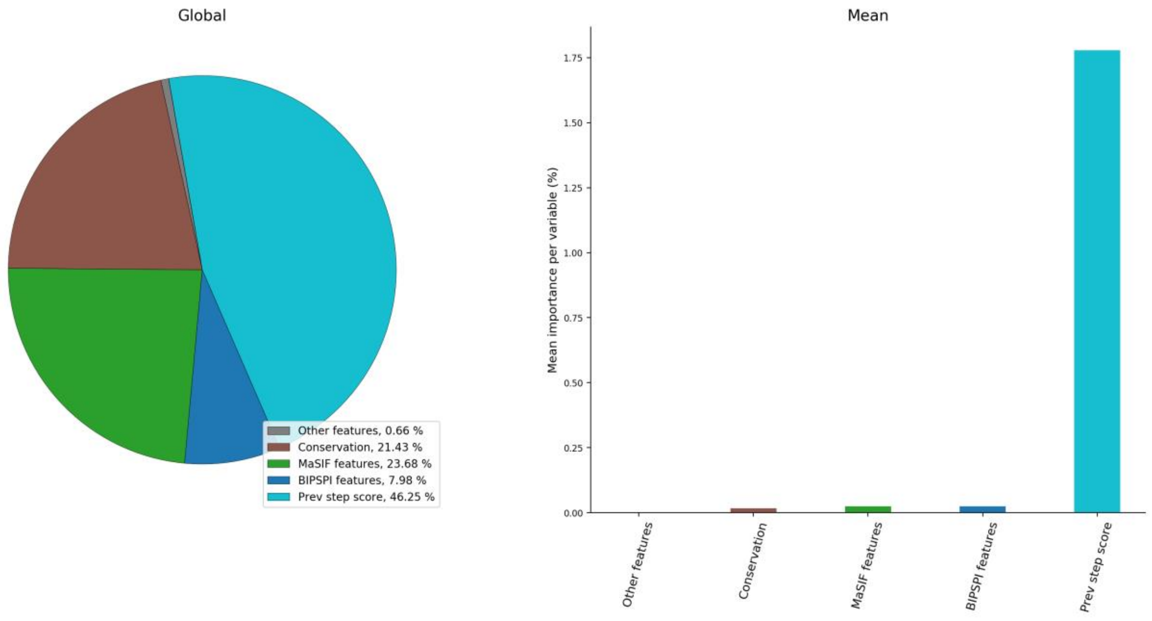

Figure 35: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-default-patch*. The structural features associated with BIPSPI and MaSIF are grouped into two separate categories called 'BIPSPI features' and ‘MaSIF features’.

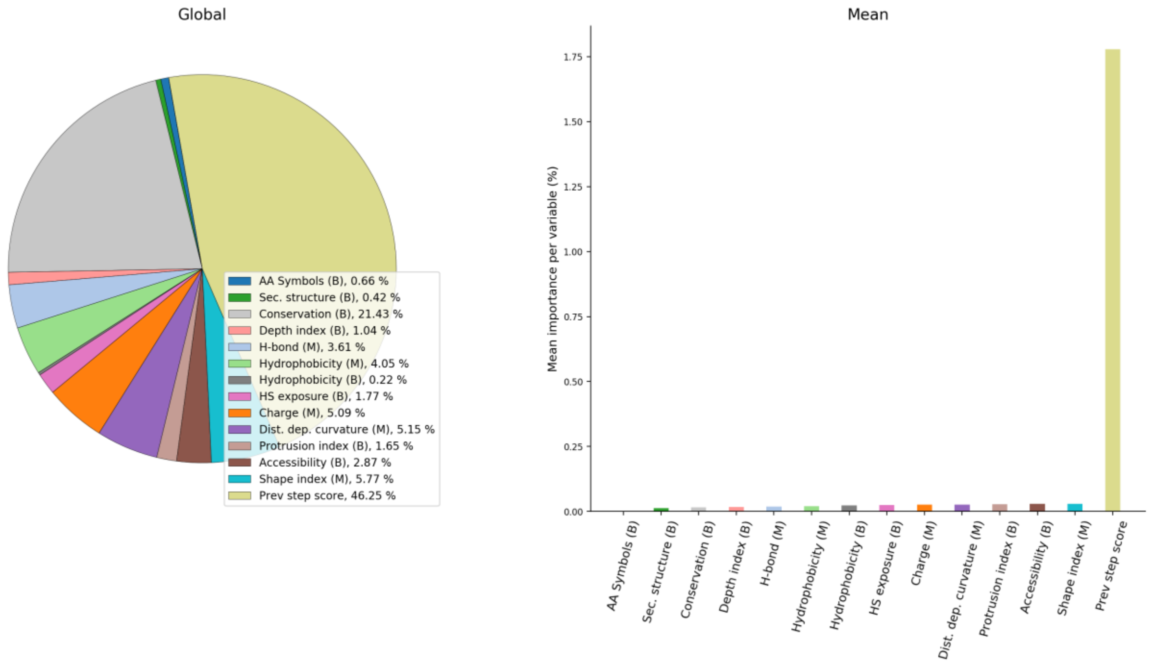

Figure 36: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-default-patch*

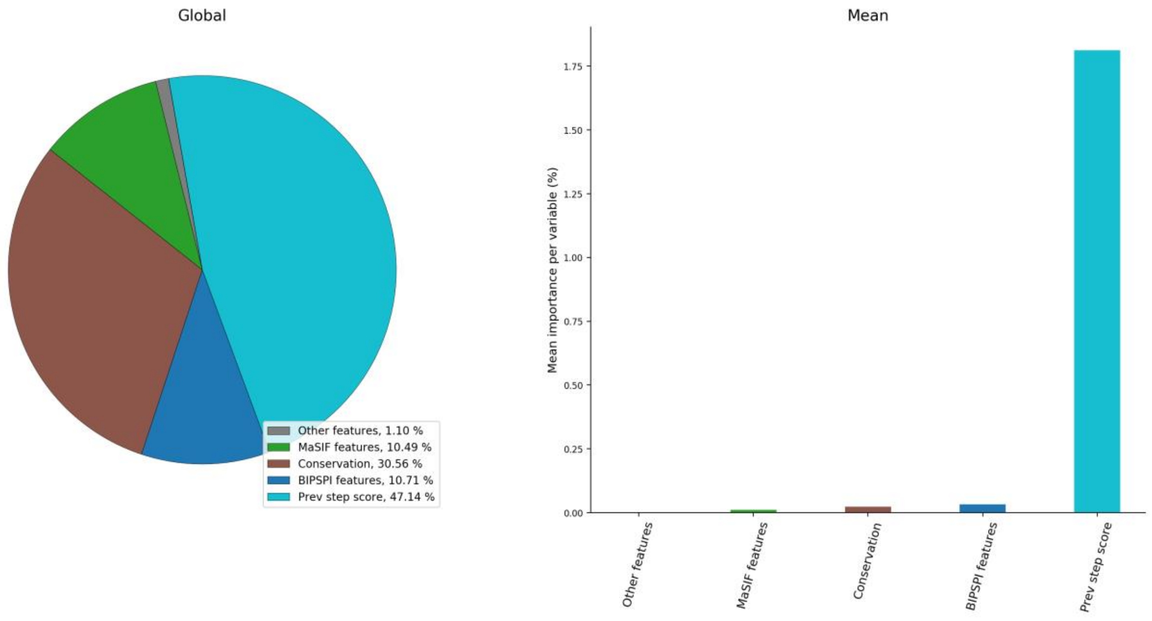

Figure 37: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-default-patch-sorted*. The structural features associated with BIPSPI and MaSIF are grouped into two separate categories called 'BIPSPI features' and ‘MaSIF features’.

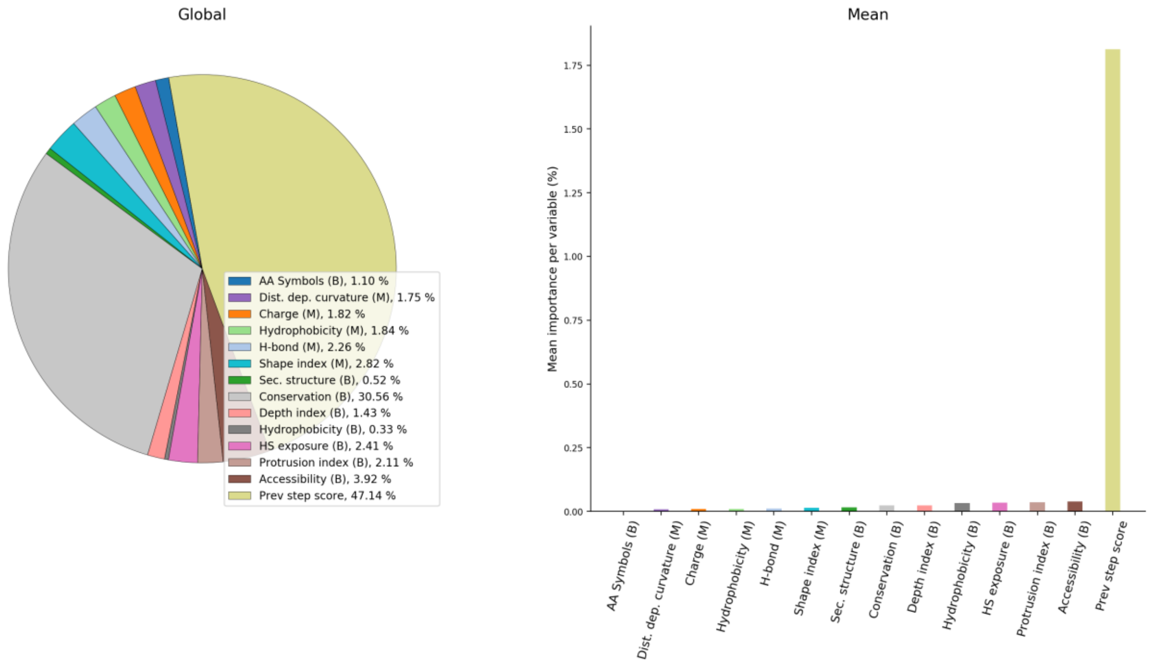

Figure 38: Global and mean feature importances (gain) associated with the features used in the training of *BIPSPI-default-patch-sorted*

Feature importance in terms of frequency

In addition to the feature importance computed in terms of ‘gain’ (presented in the main text), we also compute feature importance in terms of ‘frequency’. Here, the relative importance of a feature is computed as the ratio of the number of times a given feature is involved in tree splits across all trees that constitute the model to the total number of splits contained across all trees in the model. Since it only relies on the number of times a given feature is used, it is not as informative as ‘gain’ in relaying the impact of the features used in the various models.

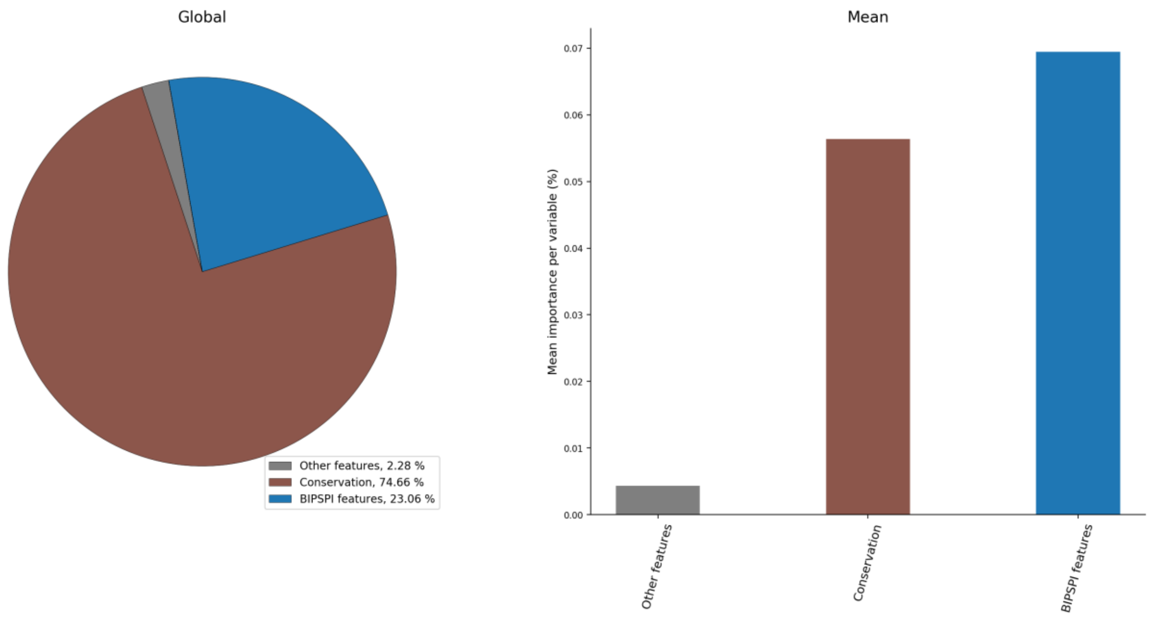

Figure 39: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-default*. The structural features associated with BIPSPI are grouped together in a single category called ‘BIPSPI features’.

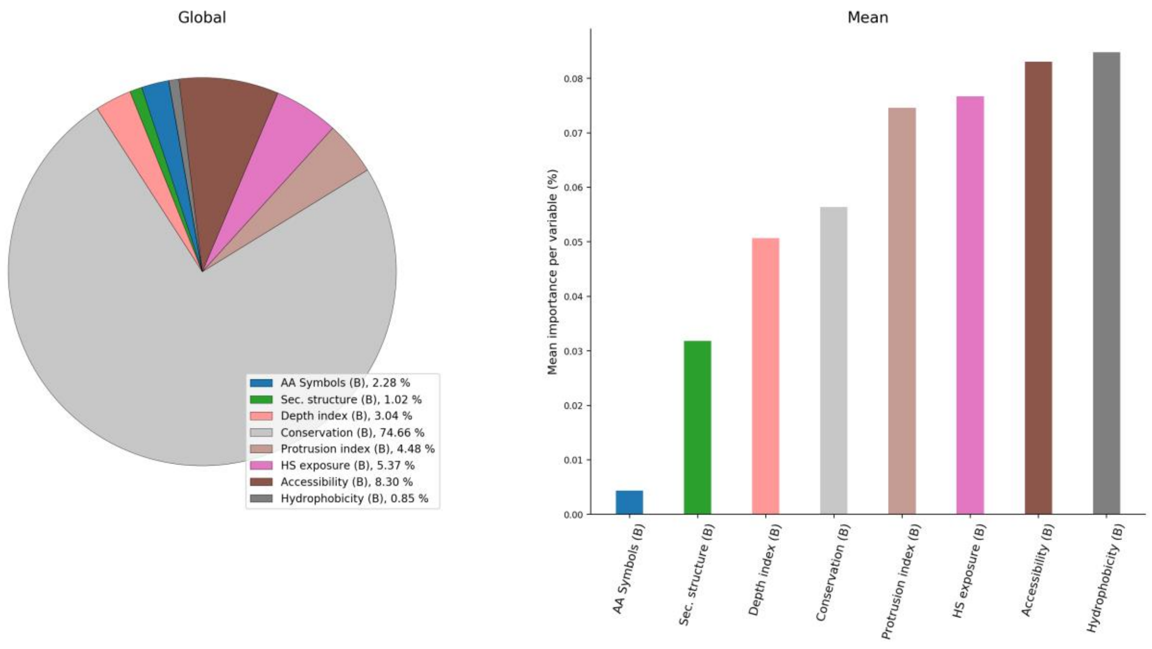

Figure 40: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-default*

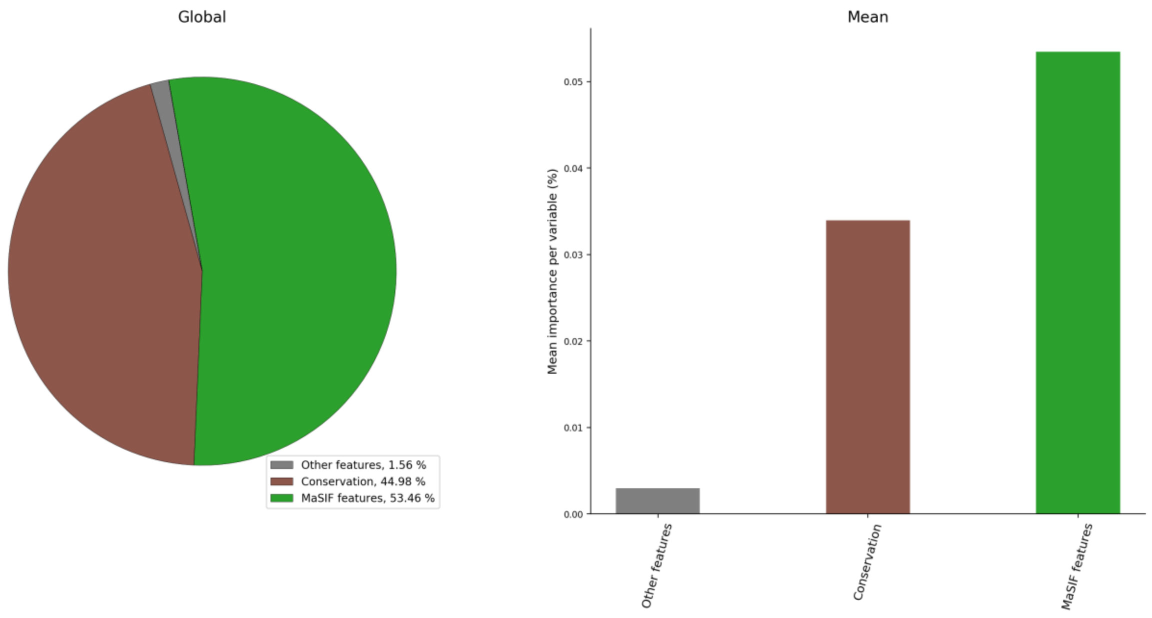

Figure 41: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-patch*. The structural features associated with MaSIF are grouped together in a single category called ‘MaSIF features’.

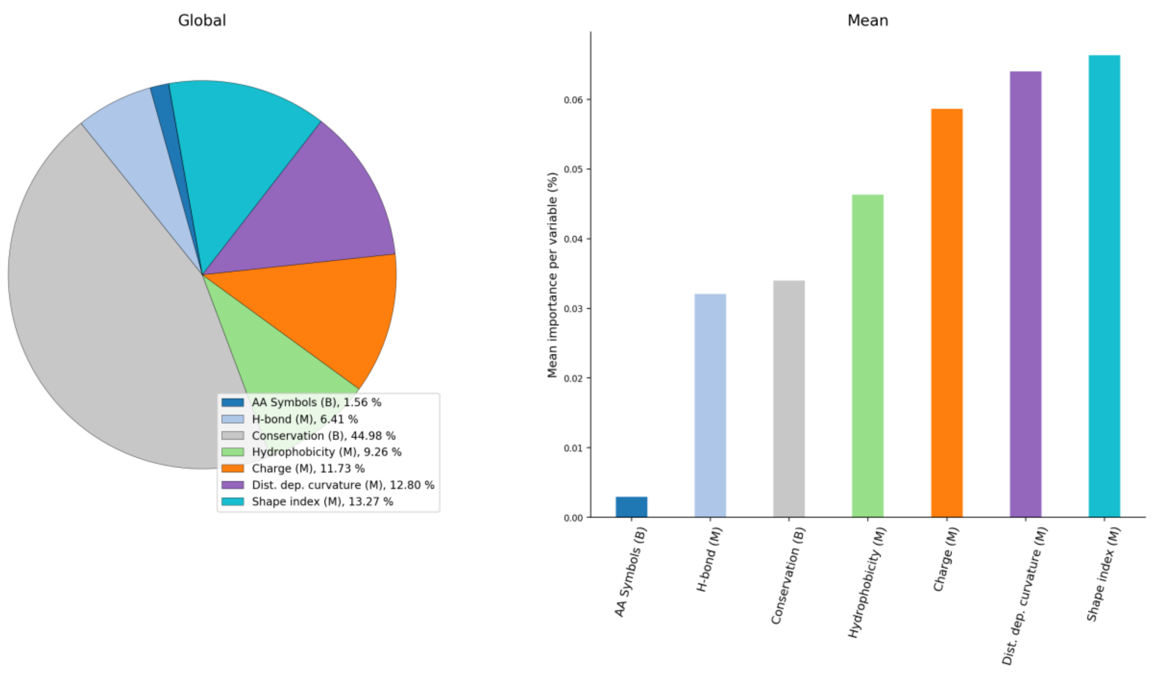

Figure 42: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-patch*

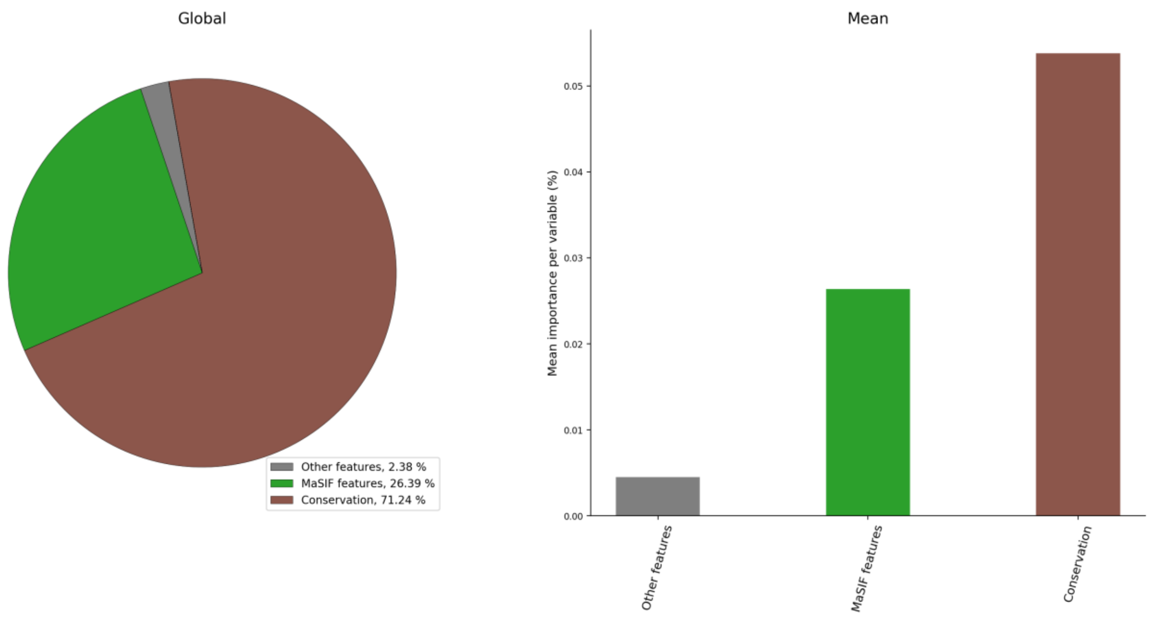

Figure 43: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-patch-sorted*. The structural features associated with MaSIF are grouped together in a single category called ‘MaSIF features’.

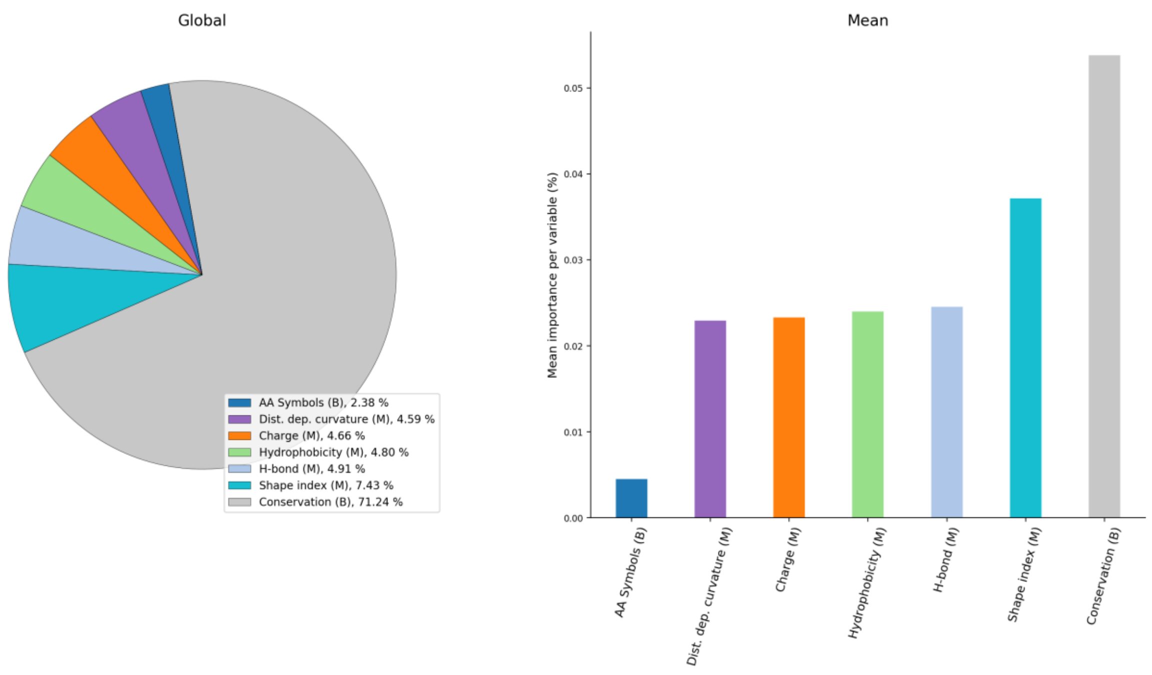

Figure 44: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-patch-sorted*

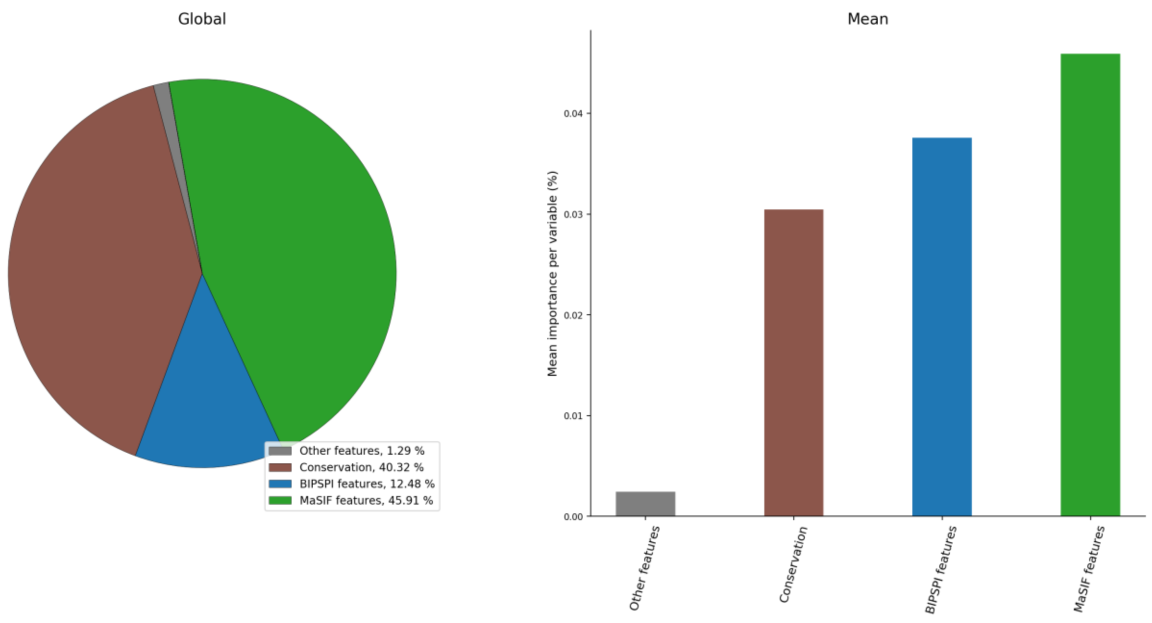

Figure 45: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-default-patch*. The structural features associated with BIPSPI and MaSIF are grouped into two separate categories called 'BIPSPI features' and ‘MaSIF features’.

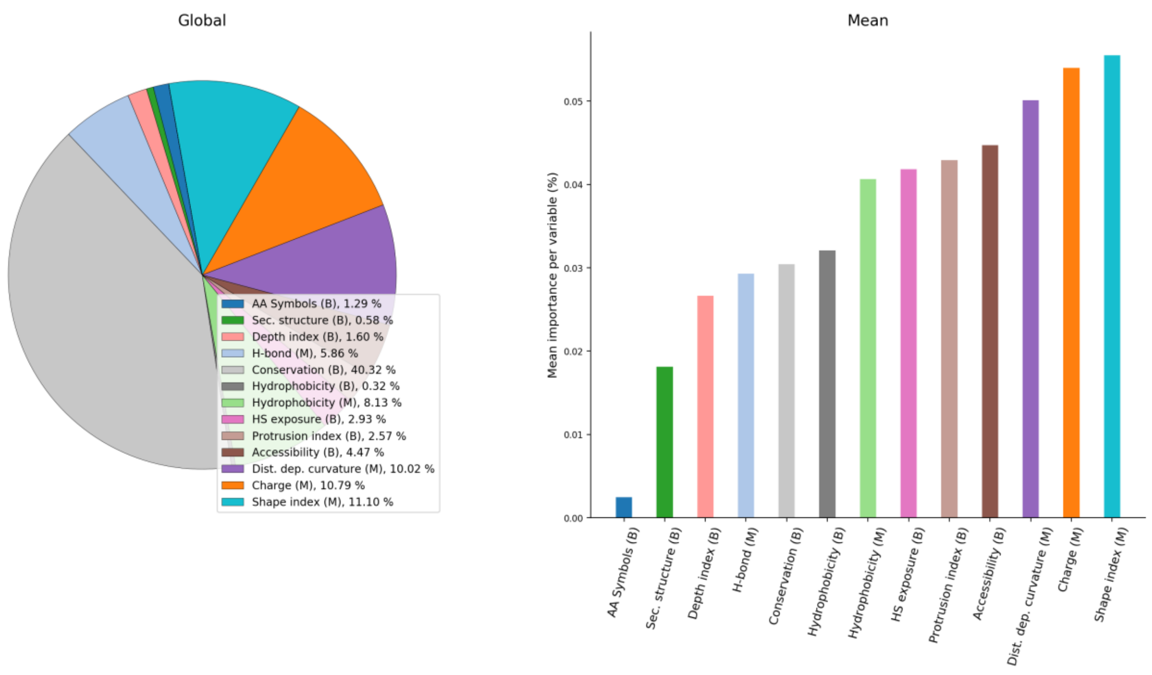

Figure 46: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-default-patch*

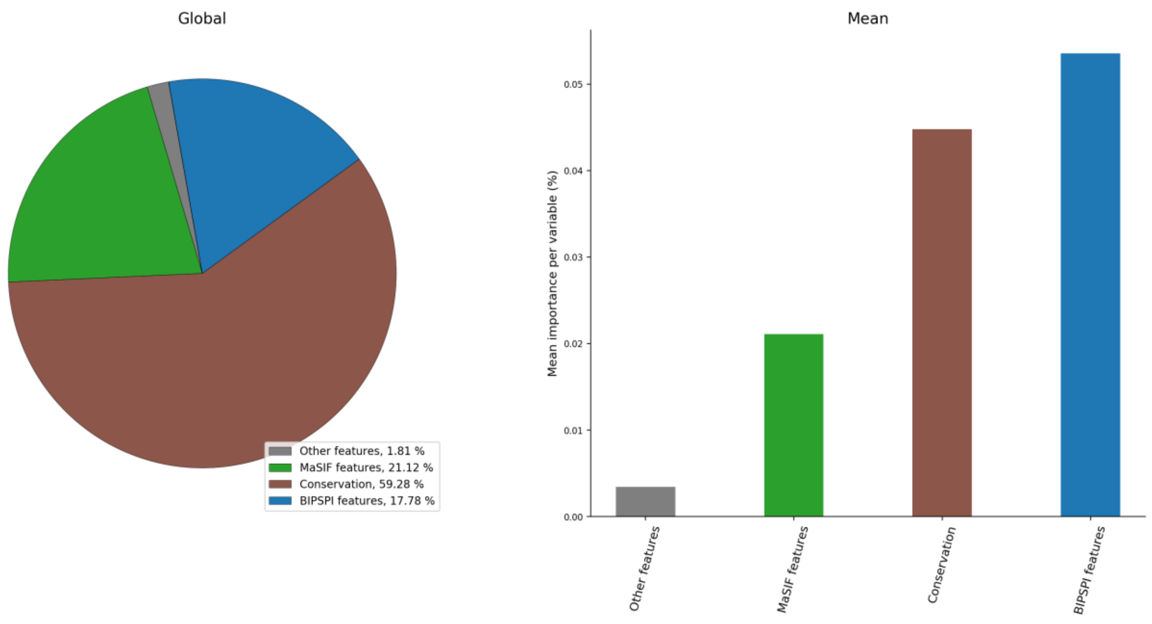

Figure 47: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-default-patch-sorted*. The structural features associated with BIPSPI and MaSIF are grouped into two separate categories called 'BIPSPI features' and ‘MaSIF features’.

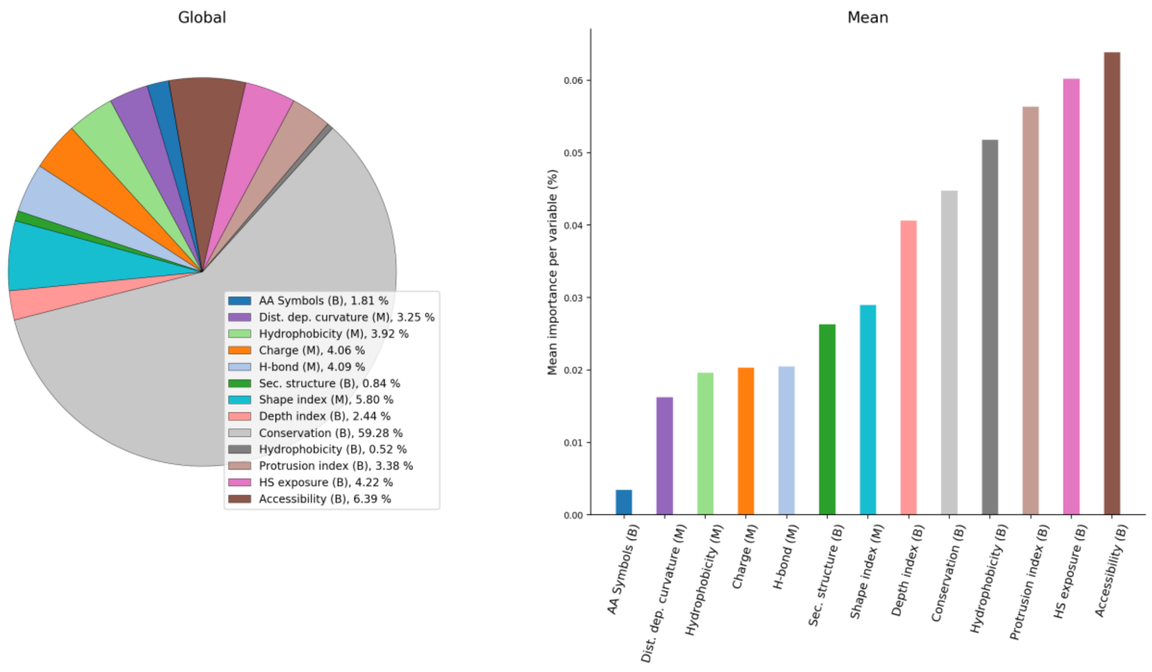

Figure 48: Global and mean feature importances (frequency) associated with the features used in the training of *BIPSPI-default-patch-sorted*