SolXplain: An Explainable Sequence-Based Protein Solubility Predictor

Raghvendra Mall

Qatar Computing Research Institute
Hamad Bin Khalifa University, Doha, Qatar

Abstract

Motivation: Protein solubility is a property associated with protein expression and is a critical determinant of the manufacturability of therapeutic proteins. It is thus imperative to design accurate *in-silico* sequence-based solubility predictors.

Methods: In this study, we propose SolXplain, an extreme gradient boosting machine based protein solubility predictor which achieves state-of-the-art performance using physio-chemical, sequence and novel structure derived features from protein sequences. Moreover, SolXplain has a unique attribute that it can provide explanation for the predicted class label for each test protein based on its corresponding feature values using SHapley Additive exPlanations (SHAP) method.

Results: Based on an independent test set, SolXplain outperformed other sequence-based methods by at least 2% in accuracy and 2% in Matthew’s correlation coefficient, with an overall accuracy of 78% and Matthew’s correlation coefficient of 0.56. Additionally, for fractions of exposed residues (FER) at various residual solvent accessibility (RSA) cutoffs, we observed higher fractions to associate positively with protein solubility, and tripeptide stretches that contain one isoleucine and one or more histidines, to associate negatively with solubility. The improved prediction accuracy of SolXplain enables it to predict protein solubility with greater consistency and screen for sequences with enhanced manufacturability.

1 Introduction

Protein solubility is an essential physiochemical property that determines the success of many biotechnical and scientific applications, including the production of therapeutic proteins and drugs based on antibodies. However, many proteins when expressed with a standard production procedure in *Escherichia coli*, have low solubility, which reduces their manufacturing capability. Enhancement of protein solubility in experimental settings is usually achieved through the use of low temperatures, weak promoters, modified growth media, or optimization of expression conditions [1, 2].

The primary determinant of protein solubility are the amino acid sequences in a protein. Previous studies showcased several correlations between amino acid sequence properties and protein solubility, such as the content of turn-forming and charged residues, the quantity of hydrophobic stretches, content of different types of residues, or the length of the protein sequence [3, 4, 5, 6].

This fact led to the development of many *in-silico* bioinformatics tools predicting the solubility propensity of proteins from their primary structure. Their aim is to replace the costly experimental procedures by pre-selecting the most promising protein sequences *in-silico*. Several protein solubility prediction tools include PROSO [7], SOLpro [2], CCSOL [8], PROSO II [9], PaRSnIP [10] and DeepSol [11]. Majority of these methods follow a two-step process: a) feature engineering and selection; b) protein solubility propensity prediction i.e. distinguishing soluble proteins from insoluble ones. PROSO tool uses SVM with Gaussian kernel and a Naive Bayes classifier to build its classifier. SOLpro utilizes a two-stage SVM with sequential minimal optimization to build the protein solubility predictor. CCSOL uses an SVM classifier and identifies coil/disorder, hydrophobicity, β -sheet and α -helix propensities as most discriminative features. PROSO II method uses a Parzen window model with modified Cauchy kernel and a two-level logistic classifier. PaRSnIP uses a gradient boosting machine algorithm which provides feature importance during training

phase. Recently, a deep learning technique called DeepSol [11], showcased that by just using the raw protein sequences as input, it can perform as well as its competitors. Furthermore, by augmenting few additional physio-chemical features, it can easily surpass all other *in-silico* sequence-based protein solubility predictors. However, it is difficult to determine the biological relevance of the features engineered by deep learning as mentioned in [11, 12].

To overcome the limitations of existing methods, we propose SolXplain, an XGBoost based model [13] using several well-known physio-chemical and sequence derived features. Additionally, SolXplain uses several novel secondary structure, disorder, disulfide bonds related, torsion angles and contact number related features extracted from the SCRATCH suite [14], DISOPRED version 3.16 [15], DISpro [16] and SPIDER3 [17] respectively. XGBoost is an optimized version of gradient boosting machine [18], which has been shown to perform very well on several bioinformatics problems, such as gene regulatory network reconstruction [19, 20, 21], transmembrane protein solubility [22] etc. Moreover, SolXplain has a unique attribute that can provide explanation for the predicted class label for each test protein based on its corresponding feature values using SHapley Additive exPlanations (SHAP) [23] algorithm. Our primary contributions include:

1. Engineering of novel structure, disorder, disulfide bonds, torsion angles and contact number related features from the protein sequence using SCRATCH suite, DISOPRED, DISpro and SPIDER3 respectively.
2. Usage of an XGBoost model enables SolXplain to outperform existing methods for various evaluation metrics on an independent test set.
3. Provide explanation for SolXplain's output by showing the most important features driving the model predictions towards soluble or insoluble class.

Figure 1 provides the steps undertaken by the proposed SolXplain model.

Figure 1: Flowchart followed by the proposed SolXplain model.

2 Materials and Methods

2.1 Overview

Our task is a binary classification problem i.e. distinguishing soluble proteins from insoluble ones. Our goal is to learn a function (H) which takes features engineered from a protein sequence i.e. $\mathbf{x} \in \mathbb{R}^d$ as input, and outputs a prediction score between $[0, 1] \in \mathbb{R}$ i.e. $H : \mathbf{x} \rightarrow [0, 1]$. In this work, H is an XGBoost model [13], a white-box non-linear tree-based interpretable boosting machine that exploits the interactions between the engineered features.

Figure 2: Different sets of features engineered for the SolXplain model. The number of features for each component is shown within parentheses.

2.2 Data Information

The original PROSO II dataset, consisting of 58,689 soluble and 70,954 insoluble sequences, was utilized as the training set. An independent test set of 1,000 soluble and 1,001 insoluble protein sequences first described by [24] was used as benchmark test set to evaluate the performance of SolXplain in comparison to other bioinformatic predictors.

We performed two main pre-processing steps to ensure sequence diversity within the training set and between training and independent test set. First, CD-HIT [25, 26] was used to reduce sequence redundancy within the training data set with a maximum sequence identity of 90%. Second, we excluded all training set sequences with a sequence identity of 30% or greater to any sequence in the independent test set to reduce the bias introduced by homologous sequences. The final training data set had 28,972 soluble and 40,448 insoluble sequences.

2.3 Feature Engineering

One of our primary contributions is to extract and design novel features which are useful for discriminating soluble proteins from insoluble ones. We devise three groups of features which are then used to train the SolXplain model (see Figure 2). The first set is composed of features based on global properties of the protein including sequence length ($\log(L)$), molecular weight ($\log(\text{Mol-Weight})$), fraction of turn-forming residues, the average of hydrophobicity (Gravy) and aliphatic indices, along with the total absolute charge. The second group of features are derived directly from the protein sequence and consist of frequencies of mono- (single amino acids, denoted AA), di- (two consecutive AAs) and tri-peptides (three consecutive AAs) within the protein sequences.

The third group consists of structural features obtained from the protein sequence using SCRATCH [14], DISOPRED [15], DISpro [16] and SPIDER3 [17]. It has been shown previously [27] that SCRATCH based features are useful for protein fold prediction. We predict 3- and 8-state secondary structure (SS) information as well as the fraction of exposed residues (FER) at 20 different relative solvent accessibility (RSA) cutoffs ($\geq 0\% - \geq 95\%$ cutoffs at 5% intervals). From the 3-state SS obtained via SCRATCH, we extract mono- (1 state i.e. turn, strand or coil), di- (two consecutive states) and tri-state (three consecutive states) frequencies for a given protein sequence. We follow a similar procedure for the more granular 8-state SS information as shown in Figure 2. Additionally, we multiply the FER by the average hydrophobicity of these exposed residues at different RSA cutoffs.

From DISOPRED, we obtain information about which AAs in the protein sequences are part of disordered

regions as well as which AAs from the protein binding sites (PBS) of a protein are part of disordered regions. Given this information, we design features such as the fraction of the protein sequence which is disordered, frequency of each of the AAs (out of the 20 AAs) in disordered regions, number of disordered regions, number of disordered regions of length < 5 AAs, number of disordered regions of length between 5 and 10 AAs, and number of disordered regions of length > 10 AAs in a protein sequence. A similar set of features are extracted from the PBS of a protein.

From DISpro, we obtain information about cysteine disulfide bonds in the protein sequence. Since, the disulfide bonds are strong bonds, they are important for the structure and stability of proteins. We engineer features such as number of cysteine residues in protein sequence, ratio of number of cysteine bonds to ideal number of possible cysteine bonds, fraction of cysteine bonds between two residues which are less than 12 amino acids away from each other i.e. fraction of small bonds, fraction of cysteine bonds between two residues which are greater than 12 AA but less than 24 AA away from each other i.e. fraction of medium bonds and finally fraction of cysteine bonds between two residues which are greater than 24 AAs away from each other i.e. fraction of large bonds.

From SPIDER3, we extract features related to torsion angles (ψ and ϕ), half sphere exposure (HSE) and contact number (CN) which are important to determine the structure and stability of a protein. We engineer 4 features determining density of residues in α -helix, β -sheet, lefthand α and the remaining regions from the Ramachandran plot [28] based on the ψ and ϕ angles predicted by SPIDER3. Higher fraction of residues in α -helix, β -sheet and lefthand α regions of the Ramachandran plot are characteristics of a stable protein structure. We finally extract 11 features related to contact numbers and HSE of residues in a protein sequence such as average contact number, HSE value of residues in the range set $S = \{\leq 10, > 10 \& < 20, \geq 20\}$, AAs away from these residues.

In total, we include 9158 features for each protein sequence. In contrast to other sequence-based predictors, we don't perform a feature selection step to exclude features, rather we rely on the XGBoost model to prioritize the most important features and filter out irrelevant ones.

2.4 Methods

2.4.1 Gradient Boosting Machine

The SolXplain model is an optimized version of the white-box, non-linear, ensemble gradient boosting machine (GBM) [18, 29] called XGBoost [13]. Gradient Boosting is a machine learning method based on a constructive strategy by which the learning procedure will additively fit new models, typically decision trees and repetitively leverage the patterns in residuals to provide a more accurate estimate of the response variable (soluble versus insoluble proteins).

2.4.2 XGBoost Algorithm

Tree boosting is a learning technique to improve the classification of weaker classifiers by repeatedly adding new decision trees to the ensembles. XGBoost [13] is a scalable machine learning technique for tree boosting. It was shown in [13] that its performance is better than other boosting algorithms.

The main components of XGBoost algorithm are the objective function and its iterative solution. The objective function is initialized to describe the model's performance. Given the training dataset, $\mathbb{D}_1 = \{x^i, y^i\}_{i=1}^N$ where $x^i \in \mathbb{R}^d$, $d = 9158$, $y^i \in \mathbb{R}$ and N denotes the number of total number of training samples. The predicted output $\hat{y}^i$ obtained from the ensemble model can be represented as: $\hat{y}^i = \sum_{t=1}^T H_t(x^i)$, where $H_t(x^i)$ represents the prediction score of the t^{th} decision tree for the i^{th} protein sequence in the training dataset. If the decision trees are allowed to grow unregulated, then the resulting model is bound to overfit [13]. Hence, the following objective has to be minimized:

$$\mathbb{J}(H) = \sum_{i=1}^N \mathbf{L}(y^i, \hat{y}^i) + \sum_{t=1}^T \Omega(H_t), \quad (1)$$

where $\mathbf{L}$ is the loss function and $\Omega(\cdot)$ is the penalty that is used to prevent overfitting which is defined as: $\Omega(H_t) = \gamma A + \frac{1}{2} \lambda \sum_{j=1}^A w_j^2$, where γ and λ are the parameters which control the penalty for the number of leaf nodes (A) and leaf weights (w) respectively in the decision tree H_t .

The objective function can be re-written as follows: $\mathbb{J}(H_t) = \sum_{i=1}^N \mathbf{L}(y^i, \hat{y}_{t-1}^i + H_t(x^i)) + \Omega(H_t)$. After applying a Taylor expansion and expanding $\Omega(H_t)$, we obtain:

$$\mathbb{J}(H_t) = \sum_{i=1}^N [g_i H_t(x^i) + \frac{1}{2} h_i^2 H_t(x^i)] + \gamma A + \frac{1}{2} \lambda \sum_{j=1}^A w_j^2,$$

where $g_i = \partial_{\hat{y}_{t-1}}(\mathbf{L}(y^i, \hat{y}_{t-1}^i))$ and $h_i = \partial_{\hat{y}_{t-1}}^2(\mathbf{L}(y^i, \hat{y}_{t-1}^i))$ are the first and second order gradient statistics on the loss function. For a fixed tree structure $H(\mathbf{x})$, where $I_j = \{i | H(x^i) = j\}$ is an instance of leaf node j , the optimal weight w_j^* for leaf node j is:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

The corresponding optimal objective function then becomes:

$$\mathbb{J}(H_t) = -\frac{1}{2} \sum_{j=1}^A \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma A \quad (2)$$

Equation 2 can be used as a scoring function to measure the quality of a tree structure H_t at the t^{th} iteration. This score is equivalent to the impurity score used for evaluating decision trees in random forests [30]. The authors in [13], devise a fast, greedy and iterative algorithm to identify these optimal tree structures.

2.5 Training

We train our XGBoost classifier on top of physio-chemical (global), sequence and structural features extracted from the protein sequence as mentioned earlier. Since our training set has slight imbalance, we weigh the samples belonging to soluble class by α , where $\alpha = \frac{N_p}{N_n}$ and N_n is the total number of insoluble proteins and N_p is the total number of soluble proteins in the training set.

The SolXplain classifier is based on several parameters, such as maximum depth of a tree (M), the learning rate (ν), the minimum child weight of a leaf node (w_j), the sampling rate for features (r), and the regularization parameter γ . We keep the sampling ratio from training set parameter to a fixed value of 0.8. We then performed a hyper-parameter optimization procedure by varying these parameters over a grid of $M \times \nu \times w_j \times r \times s = 243$ combinations. In particular with $M \in \{4, 7, 10, \dots\}$, $\nu \in \{0.01, 0.1, 0.2\}$, $w_j \in \{5, 10, 20\}$, $r \in \{0.1, 0.2, 0.3\}$, and $\gamma \in \{0.0, 0.25, 0.5\}$. We performed a ten-fold cross-validation for each of these combinations. Finally, we selected the XGBoost classifier which had the maximum ten-fold cross-validation area under the curve, which was obtained corresponding to the parameters $M = 10$, $\nu = 0.01$, $w_j = 20$, $r = 0.3$ and $\gamma = 0.0$. The final SolXplain classifier had a maximum training accuracy of 85.93% and a maximum AUC of 0.94.

2.6 Evaluation Metrics

SolXplain method was compared against different in-silico sequence-based solubility predictors using several well-known metrics such as Accuracy (ACC), Matthew's correlation-coefficient (MCC), Selectivity per class, Sensitivity per class and Gain per class. All these evaluation metrics are based on true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The set TP represents the proteins which are soluble (class label 1) and for which SolXplain predicts $H(\mathbf{x}) \geq 0.5$. Similarly, the set TN consists of those proteins which are insoluble (class label 0) and for which SolXplain predicts $H(\mathbf{x}) < 0.5$. The set FP represents those proteins whose true label is insoluble i.e. 0 but SolXplain predicts $H(\mathbf{x}) \geq 0.5$ and the set FN comprises proteins whose true label is soluble i.e. 1 but SolXplain estimates $H(\mathbf{x}) < 0.5$. A detailed definition of these sets and importance of each of these evaluation metrics are provided in [11, 10, 31]. Higher the value of these metrics the better the performance of the predictor.

3 Experimental Results

3.1 Performance Comparison

The prediction performance of SolXplain was assessed using an independent test set first reported by [24]. We compared SolXplain to state-of-the-art solubility predictors DeepSol S2, DeepSol S1, PaRSnIP and PROSO II. We also implemented an optimized deep feed-forward neural network (DNNSol) on top of 9,158 features for comprehensive comparison as depicted in Figure 3. SolXplain yielded a prediction accuracy of 77.79% and an MCC of 0.560, outperforming the state-of-the-art method DeepSol S2 by at least 2% in accuracy and 2% in MCC (see Table 1).

Figure 3: SolXplain performance comparison with other state-of-the-art solubility predictors.

Method	Accuracy	MCC	Selectivity (Soluble)	Selectivity (Insoluble)	Sensitivity (Soluble)	Sensitivity (Insoluble)	Gain (Soluble)	Gain (Insoluble)
PROSO II	0.672	0.344	0.667	0.678	0.687	0.657	1.335	1.355
PaRSnIP	0.740	0.481	0.758	0.725	0.705	0.775	1.517	1.448
DeepSol S1	0.728	0.458	0.748	0.712	0.689	0.768	1.495	1.424
DeepSol S2	0.766	0.546	0.843	0.718	0.654	0.878	1.686	1.434
DNNSol	0.764	0.531	0.800	0.735	0.702	0.825	1.602	1.469
SolXplain	0.780	0.560	0.773	0.786	0.791	0.786	1.547	1.572

Table 1: SolXplain outperforms all other predictors in 5 out of 8 evaluation metrics. Best results are highlighted in bold.

SolXplain achieved stable values between 0.77 and 0.79 in sensitivity and selectivity metrics for both soluble and insoluble instances, whereas other predictors were mostly class biased. For example, the nearest competitors DeepSol S2 and DNNSol were better than SolXplain in detecting insoluble proteins as they attained high sensitivity for insoluble proteins and high selectivity for soluble proteins i.e. fewer insoluble proteins were misclassified as soluble proteins. However, DeepSol S2 and DNNSol were extremely poor, $\approx 15\%$ and $\approx 9\%$ worse than SolXplain in sensitivity (soluble class) respectively i.e. accurately identifying soluble proteins. Similarly, other classifiers such as DeepSol S1 and PaRSnIP were biased towards the insoluble class. However, their performance was worse than SolXplain for all evaluation metrics. From

Figure 3b, we observed that the optimal cut-off (θ) to categorize the prediction of each predictor to soluble or insoluble class was ≈ 0.5 .

Figure 4: Comparison of score distribution of DeepSol S2, DNNSol with SolXplain. All the three solubility predictors have very different score distributions for class 0 or the insoluble class. However, the output score distribution of DeepSol S2 is not significantly different from SolXplain for the soluble class ($p = 0.28$). Here 'p' represents p-values and 'yellow' diamond corresponds to the mean of each score distribution.

By using score distribution plots as depicted in Figure 4, we were able to compare DeepSol S2 with DNNSol and our proposed approach SolXplain. It can empirically be illustrated that the score distributions for the 3 methods did not follow a normal distribution, since the violin plots don't mimic a gaussian distribution. Hence, we used the Mann-Whitney-Wilcoxon (MWW) test [32] to compare pairwise score distribution for each class.

We found that the score distributions of DeepSol S2 vs. DNNSol, DNNSol vs. SolXplain and DeepSol S2 vs. SolXplain were all different ($pvalue < 1e-4$) in the case of the insoluble class and the difference between the mean scores for DeepSol S2 vs. DNNSol and DNNSol vs. SolXplain were also statistically significant ($pvalue < 1e-3$) for the soluble class. Additionally, from the violin plot, we observed that the score distributions of DeepSol S2 and DNNSol have nearly similar shapes, indicative of both belonging to the deep learning class of machine learning methods, but very different from the density distribution achieved by SolXplain. This indicated that the underlying mechanism being used by SolXplain for identifying insoluble proteins was disparate from that being used by DeepSol S2 and DNNSol. However, the difference between the mean score for DeepSol S2 vs. SolXplain was not significantly different ($pvalue = 0.28$) even though their density distribution were quite dissimilar.

3.2 Feature Importance via XGBoost

An advantage of tree-based non-linear machine learning techniques, in contrast to black-box modelling techniques like support vector machines [33] and artificial neural networks [34], is that we can easily obtain feature/variable importance scores for all input features. The importance of a feature is the sum of information gained when splits (tree branching) are performed using that variable. A distinct benefit of using an XGBoost classifier is that out of all the 9158 features used during training, variables which are not used for optimal tree splits in the SolXplain model are pruned automatically and get a feature importance score of 0. We observe from Table 2 that only 499 features have non-zero feature importance scores.

We then analyzed the total feature importance contribution of all features according to their feature types/classes as shown in Figures 2 and Table 2. At the highest level, we had three macro classes of features including global, sequence, and structure derived features contributing 0.61%, 52.86%, and 46.53% respectively in the overall variable importance scores. From Table 2, we can observe that the maximum feature importance is associated with tripeptide frequencies (36.62%) followed by FER at different RSA cutoffs (20.17%). It is noteworthy that out of the 8,000 features associated with tripeptide frequencies only 72 have non-zero feature importance. Thus, majority of the non-essential features are automatically pruned by XGBoost model. Figure 5 showcases the difference between the feature values for these top 3 variables in case of soluble versus insoluble proteins.

Figure 5: Distribution of the feature values of top 3 variables discriminating soluble from insoluble training protein sequences shown as box plots (**** : P-value < 1e-4).

3.3 Model Explanation via SHAP method

A disadvantage of the inherent feature importance scores obtained from the XGBoost model is that the directionality is not apparent i.e. when a particular feature for a protein sample takes a high/low value, does it correspond to high/low feature importance. Moreover, at the test phase, it is not straightforward for white-box tree-based machine learning techniques to provide information about, say, the top k features driving the prediction to be soluble or insoluble class.

Recently, several techniques such as the LIME [35] and SHAP [23] methods have been proposed to overcome the aforementioned limitations. These methods have the ability to interpret feature importance scores from complex training models as well provide interpretable predictions for a test sample by grounding their reasoning on the top k features for that particular test instance. In our work, we use the SHAP

Feature Class	Total Features	Total Features (> 0)	Total Imp (%)
Log(L)	1	1	0.18
Log(MW)	1	1	0.18
Turn Freq	1	1	0.06
Gravy Index	1	0	0.00
Aliphatic Index	1	1	0.09
Total Charge	1	1	0.10
Mono Freq	20	19	2.05
Di Freq	400	200	14.20
Tri Freq	8000	72	36.62
Mono Freq SS3	3	3	0.24
Di Freq SS3	9	8	0.57
Tri Freq SS3	27	15	0.91
Mono Freq SS8	8	7	0.86
Di Freq SS8	64	22	2.13
Tri Freq SS8	512	69	4.66
FER at RSA cutoffs	20	18	20.17
FER at RSA cutoff $\times$ HP	20	17	11.71
Disorder Features	25	17	1.95
PBS in Disorder Features	25	10	1.86
Disulfide Features	5	4	0.33
Torsion Angle Features	4	3	0.13
Contact Number Features	10	10	1.01
All Features	9158	499	100

Table 2: Variable importance percentages grouped by feature classes for the SolXplain model highlighting the contributions of features as depicted in Figure 2. Here total features (> 0) represents the number of variables from a feature class with > 0 variable importance scores.

(SHapley Additive exPlanations) method, a unified framework for interpreting predictions, as it was shown in [23] to outperform LIME and demonstrate that its predictions are better aligned with human intuitions.

The SHAP method belongs to the class of additive feature attribution methods where a test instance prediction is composed as a linear function of features and satisfies 3 critical properties including local accuracy, missingness and consistency. The explicit SHAP regression values comes from a game-theory framework [36, 37] and can be computed as:

$$\phi_i = \sum_{S \subseteq Q \setminus \{i\}} \frac{|S|!(|Q| - |S| - 1)!}{|Q|!} [H_{S \cup \{i\}}(x_{S \cup \{i\}}) - H_S(x_S)]$$

Here Q represents the set of all d features and S represents the subsets obtained from Q after removing the i^{th} feature and ϕ_i is an estimate of the importance of feature i in the model. In order to refrain from undergoing $2^{|Q|}$ differences to estimate ϕ_i , the SHAP method approximates the Shapley value by either performing Shapley sampling [38] or quantitative input influence [39]. A detailed description of the SHAP method for model interpretation is available in [23].

We passed our SolXplain model along with the training set features to the SHAP method as shown in Figure 1 to obtain importance of features based on Shapley values. Figure 6 highlights the top 20 training features based on Shapley values. Moreover, it also provides directionality i.e. when a feature attains “high” or “low” values, the corresponding Shapley values are positive or negative. The positive Shapley values drive the predictions towards soluble class whereas the negative Shapley values influence the predictions to move towards the insoluble class. From Figure 6, we can observe that when top features such as FER at RSA cutoffs $\geq 75\%$, $\geq 70\%$, and $\geq 60\%$ take high values, the corresponding Shapley values are positive driving model prediction to soluble class, whereas when these features take low values (i.e. closer to 0), the corresponding Shapley values are negative. Similarly, Figure 6 illustrates that when top features such as

Figure 6: Top 20 features from SHAP [23] method. Here 'gteq' refers to $\geq$.

tripeptides: IHH and KIH frequencies are high, the corresponding Shapley values are negative and driving the model prediction to the insoluble class.

3.4 Case Study

To better understand the varying SolXplain predictions, we analyzed the top 5 positive and top 5 negative features obtained via SHAP algorithm for two sample proteins. *Acinetobacter baylyi* pyrimidine nucleoside phosphorylase (3HQX) attained a relatively high SolXplain prediction score of 0.935, while the human AT-rich interactive domain-containing protein 3A's (ARID3A) got a score of 0.725.

The first protein's top 5 positive features were almost exclusively made up of FER at different RSA cutoffs and had high positive Shapley values whereas the top 5 negative features had very small negative Shapley values. This drove the model prediction to be very close to 1 (0.935), suggesting the protein belonged to soluble class. We could observe from Figure 7 A and B, that there were several sequence of residues which were solvent accessible even at higher RSA cutoffs. For the second example protein again, the FER residues at different RSA cutoffs were the primary driver towards protein solubility prediction and attained high positive Shapley values. However, presence of tripeptide such as methionine-glycine-histidine and histidine-serine-histidine had a high negative Shapley value as depicted in Figure 7F. Thus, SolXplain achieved a

relatively lower solubility score for ARID3A in comparison to 3QHx protein. Finally, from Figure 7 B and E, it was readily apparent that the phosphorylase had more RSA amino acids (depicted with light colors) than the second example protein ARID3A. Furthermore, ARID3A had 2 disordered regions at its terminal regions, which were not solved in the crystal structure, illustrated by dashed lines (7 D), that had antagonistic effect on the protein solubility.

Figure 7: Top predictive features - relative solvent accessibilities and tripeptide frequencies - correlate with SolXplain scores. (A) Structure of pyrimidine/purine nucleoside phosphorylase with predicted RSA values mapped on structure in cartoon illustration (PDB ID: 3HQX, 108 residues). (B) Predicted RSA values shown on structure in surface representation. (C) Top 10 SHAP-features with corresponding Shapely values. (D) Structure of human AT-rich interactive domain-containing protein 3A (ARID3A, 145 residues) in cartoon illustration. (E) Predicted RSA values mapped onto structure and shown in surface representation. (F) Top 10 SHAP-features with corresponding Shapely values.

4 Discussion

The development of *in silico* sequence-based protein solubility prediction tools with high accuracy continues to be highly sought after. In this study, we introduce SolXplain, a solubility predictor that uses the XGBoost modeling technique and features that represent physio-chemical, sequence as well as structural properties of proteins. SolXplain outperforms, to the best of our knowledge, several existing sequence-based solubility predictors by > 2% in accuracy and > 2% in MCC.

The superiority of SolXplain over other predictors is due to three factors. The first factor is the choice of the machine learning method XGBoost. The non-linear optimized gradient boosting technique, XGBoost,

is able to capture non-linear relationships between the features and the dependent vector, which makes its performance comparable to non-linear methods like SVMs [40]. Additionally, XGBoost reduces the bias of the model without increasing the variance, leading to better generalization performance. In addition, XGBoost has the ability to provide variable importance, making the model interpretable, which is a drawback of black-box methods like SVMs and deep learning [41]. The second factor is the choice of features. We include several features that provide information about the physio-chemical, sequence and structural properties of the protein of interest. Previous tools such as DeepSF [27] have shown that features extracted from SCRATCH suite are very helpful in correct protein fold recognition. We observe that the FER at different RSA cutoffs, average hydrophobicity of such residues and tripeptide frequencies play a very vital role in protein solubility prediction. An inherent advantage of XGBoost model is that it performs regularization i.e. feature pruning automatically, reducing the risk of overfitting and including only those features which help in discriminating the positive class from the insoluble ones. Thus, it has an advantage over two-stage methods like PROSO II and Solpro, which are susceptible to loss of information by explicitly performing feature selection. Finally, unlike other sequence-based predictors, SolXplain has the ability to provide a meaningful explanation for each test sample using the SHAP method. This empowers biologists to quickly screen for good solubility targets and to attempt mutations of initial targets for soluble protein production reliably and with more information to reason about the effects of proposed modifications.

From the SolXplain model, (see Figure 5), we observe that the features with the highest variable importance were FER at RSA cutoffs 75%, FER at RSA cutoffs 65% $\times$ by average hydrophobicity of these residues and tripeptide IHH frequency. We notice from the training set that the FERs for the soluble set is significantly higher than the FERs for the insoluble set (see Figure 5, P-value $< 1e-4$), which is the reason that the FER is a dominant feature of the classifier. Additionally, from Figure 6, we also detect that the insoluble proteins tend to have higher frequency of tripeptides containing one isoleucine and one or more histidines in the protein sequence whereas soluble proteins have higher frequency of tripeptides containing only histidines. Interestingly, positively charged surface residues and polyhistidine-tags have been previously [42] correlated with protein solubility.

In this work, we develop SolXplain, a novel sequence-based protein solubility predictor that uses XGBoost technique and a large set of features depicting physio-chemical, sequence and structural properties of proteins. SolXplain not only outperforms all existing sequence-based protein solubility predictors, but is the first approach that provides meaningful interpretations for its prediction by using the SHAP method. SolXplain has relatively high sensitivity for soluble class (0.791) suggesting that it can efficiently select soluble proteins from an initial set of candidates, thereby, reducing the high attrition rate and the production cost.

References

- [1] S. Idicula-Thomas, P. V. Balaji, Understanding the relationship between the primary structure of proteins and its propensity to be soluble on overexpression in *Escherichia coli*, *Protein science : a publication of the Protein Society* 14 (3) (2005) 582–92. doi:10.1110/ps.041009005.
URL <http://www.ncbi.nlm.nih.gov/pubmed/15689506><http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2279285>
- [2] C. N. Magnan, A. Randall, P. Baldi, SOLpro: accurate sequence-based prediction of protein solubility., *Bioinformatics (Oxford, England)* 25 (17) (2009) 2200–7. doi:10.1093/bioinformatics/btp386.
URL <http://www.ncbi.nlm.nih.gov/pubmed/19549632>
- [3] D. L. Wilkinson, R. G. Harrison, Predicting the solubility of recombinant proteins in *Escherichia coli*, *Biotechnology (Nature Publishing Company)* 9 (5) (1991) 443–8.
URL <http://www.ncbi.nlm.nih.gov/pubmed/1367308>
- [4] G. D. Davis, C. Elisee, D. M. Newham, R. G. Harrison, New fusion protein systems designed to give soluble expression in *Escherichia coli*, *Biotechnology and bioengineering* 65 (4) (1999) 382–8.
URL <http://www.ncbi.nlm.nih.gov/pubmed/10506413>
- [5] D. Christendat, A. Yee, A. Dharamsi, Y. Kluger, A. Savchenko, J. R. Cort, V. Booth, C. D. Mackereth, V. Saridakis, I. Ekiel, G. Kozlov, K. L. Maxwell, N. Wu, L. P. McIntosh, K. Gehring, M. A. Kennedy,

- A. R. Davidson, E. F. Pai, M. Gerstein, A. M. Edwards, C. H. Arrowsmith, Structural proteomics of an archaeon., *Nature structural biology* 7 (10) (2000) 903–9. doi:10.1038/82823.
URL <http://www.ncbi.nlm.nih.gov/pubmed/11017201>
- [6] P. Bertone, Y. Kluger, N. Lan, D. Zheng, D. Christendat, A. Yee, A. M. Edwards, C. H. Arrowsmith, G. T. Montelione, M. Gerstein, SPINE: an integrated tracking database and data mining approach for identifying feasible targets in high-throughput structural proteomics., *Nucleic acids research* 29 (13) (2001) 2884–98.
URL <http://www.ncbi.nlm.nih.gov/pubmed/11433035><http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC55760>
- [7] P. Smialowski, A. J. Martin-Galiano, A. Mikolajka, T. Girschick, T. A. Holak, D. Frishman, Protein solubility: sequence based prediction and experimental verification, *Bioinformatics* 23 (19) (2007) 2536–2542. doi:10.1093/bioinformatics/btl623.
URL <https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btl623>
- [8] F. Agostini, M. Vendruscolo, G. G. Tartaglia, Sequence-based prediction of protein solubility, *Journal of Molecular Biology* 421 (2-3) (2012) 237–241. doi:10.1016/j.jmb.2011.12.005.
URL <http://linkinghub.elsevier.com/retrieve/pii/S0022283611012940>
- [9] P. Smialowski, G. Doose, P. Torkler, S. Kaufmann, D. Frishman, PROSO II - a new method for protein solubility prediction, *FEBS Journal* 279 (12) (2012) 2192–2200. doi:10.1111/j.1742-4658.2012.08603.x.
URL <http://doi.wiley.com/10.1111/j.1742-4658.2012.08603.x>
- [10] R. Rawi, R. Mall, K. Kunji, C.-H. Shen, P. D. Kwong, G.-Y. Chuang, PaRSnIP: Sequence-Based Protein Solubility Prediction using Gradient Boosting Machine, *Bioinformatics* (2017) btx662arXiv:oup/backfile/content_public/journal/bioinformatics/pap/10.1093_bioinformatics_btx662/1/btx662.pdf, doi:10.1093/bioinformatics/btx662.
URL [+http://dx.doi.org/10.1093/bioinformatics/btx662](http://dx.doi.org/10.1093/bioinformatics/btx662)
- [11] S. Khurana, R. Rawi, K. Kunji, G.-Y. Chuang, H. Bensmail, R. Mall, Deepsol: a deep learning framework for sequence-based protein solubility prediction, *Bioinformatics* 34 (15) (2018) 2605–2613.
- [12] A. Elbasir, B. Moovarkumudalvan, K. Kunji, P. R. Kolatkar, R. Mall, H. Bensmail, DeepCrystal: a deep learning framework for sequence-based protein crystallization prediction, *Bioinformatics*arXiv:<http://oup.prod.sis.lan/bioinformatics/advance-article-pdf/doi/10.1093/bioinformatics/bty953/27356694/bty953.pdf>, doi:10.1093/bioinformatics/bty953.
URL <https://doi.org/10.1093/bioinformatics/bty953>
- [13] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, ACM, 2016, pp. 785–794.
- [14] J. Cheng, A. Z. Randall, M. J. Sweredoski, P. Baldi, Scratch: a protein structure and structural feature prediction server, *Nucleic acids research* 33 (suppl.2) (2005) W72–W76.
- [15] J. J. Ward, L. J. McGuffin, K. Bryson, B. F. Buxton, D. T. Jones, The disopred server for the prediction of protein disorder, *Bioinformatics* 20 (13) (2004) 2138–2139.
- [16] J. Cheng, M. J. Sweredoski, P. Baldi, Accurate prediction of protein disordered regions by mining protein structure data, *Data mining and knowledge discovery* 11 (3) (2005) 213–222.
- [17] R. Heffernan, Y. Yang, K. Paliwal, Y. Zhou, Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility, *Bioinformatics* 33 (18) (2017) 2842–2849.
- [18] J. H. Friedman, Greedy function approximation: a gradient boosting machine, *Annals of statistics* (2001) 1189–1232.

- [19] R. Mall, L. Cerulo, L. Garofano, V. Frattini, K. Kunji, H. Bensmail, T. S. Sabedot, H. Noushmehr, A. Lasorella, A. Iavarone, et al., Rgbm: regularized gradient boosting machines for identification of the transcriptional regulators of discrete glioma subtypes, *Nucleic acids research* 46 (7) (2018) e39–e39.
- [20] R. Mall, E. Ullah, K. Kunji, M. Ceccarelli, H. Bensmail, An unsupervised disease module identification technique in biological networks using novel quality metric based on connectivity, conductance and modularity, *F1000Research* 7.
- [21] R. Mall, E. Ullah, K. Kunji, F. D’Angelo, H. Bensmail, M. Ceccarelli, Differential community detection in paired biological networks, in: *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics, ACM*, 2017, pp. 330–339.
- [22] J. K. Varga, G. E. Tusnady, Tmcrys: predict propensity of success for transmembrane protein crystallization, *Bioinformatics* 34 (18) (2018) 3126–3130.
- [23] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: *Advances in Neural Information Processing Systems*, 2017, pp. 4765–4774.
- [24] C. C. H. Chang, J. Song, B. T. Tey, R. N. Ramanan, Bioinformatics approaches for improved recombinant protein production in *Escherichia coli*: protein solubility prediction, *Briefings in Bioinformatics* 15 (6) (2014) 953–962. doi:10.1093/bib/bbt057.
URL <https://academic.oup.com/bib/article-lookup/doi/10.1093/bib/bbt057>
- [25] W. Li, A. Godzik, CD-HIT: a fast program for clustering and comparing large sets of protein or nucleotide sequences., *Bioinformatics (Oxford, England)* 22 (13) (2006) 1658–9. doi:10.1093/bioinformatics/bt1158.
URL <http://www.ncbi.nlm.nih.gov/pubmed/16731699>
- [26] L. Fu, B. Niu, Z. Zhu, S. Wu, W. Li, CD-HIT: accelerated for clustering the next-generation sequencing data., *Bioinformatics (Oxford, England)* 28 (23) (2012) 3150–2. doi:10.1093/bioinformatics/bts565.
URL <http://www.ncbi.nlm.nih.gov/pubmed/23060610><http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3516142>
- [27] J. Hou, B. Adhikari, J. Cheng, Deepsf: deep convolutional neural network for mapping protein sequences to folds, *arXiv preprint arXiv:1706.01010*.
- [28] G. N. Ramachandran, Stereochemistry of polypeptide chain configurations, *J. Mol. Biol.* 7 (1963) 95–99.
- [29] R. E. Schapire, The boosting approach to machine learning: An overview, in: *Nonlinear estimation and classification*, Springer, 2003, pp. 149–171.
- [30] L. Breiman, Random forests, *Machine learning* 45 (1) (2001) 5–32.
- [31] J.-C. Lamirel, P. Cuxac, R. Mall, G. Safi, A new efficient and unbiased approach for clustering quality evaluation, in: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Springer, 2011, pp. 209–220.
- [32] H. B. Mann, D. R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, *The annals of mathematical statistics* (1947) 50–60.
- [33] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, V. Vapnik, Support vector regression machines, in: *Advances in neural information processing systems*, 1997, pp. 155–161.
- [34] L. V. Fausett, et al., *Fundamentals of neural networks: architectures, algorithms, and applications*, Vol. 3, prentice-Hall Englewood Cliffs, 1994.
- [35] M. T. Ribeiro, S. Singh, C. Guestrin, Why should i trust you?: Explaining the predictions of any classifier, in: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, ACM, 2016, pp. 1135–1144.

- [36] S. Lipovetsky, M. Conklin, Analysis of regression in game theory approach, *Applied Stochastic Models in Business and Industry* 17 (4) (2001) 319–330.
- [37] L. S. Shapley, A value for n-person games, *Contributions to the Theory of Games* 2 (28) (1953) 307–317.
- [38] E. Štrumbelj, I. Kononenko, Explaining prediction models and individual predictions with feature contributions, *Knowledge and information systems* 41 (3) (2014) 647–665.
- [39] A. Datta, S. Sen, Y. Zick, Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems, in: *2016 IEEE symposium on security and privacy (SP)*, IEEE, 2016, pp. 598–617.
- [40] C.-C. Chang, C.-J. Lin, Libsvm: A library for support vector machines, *ACM transactions on intelligent systems and technology (TIST)* 2 (3) (2011) 27.
- [41] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, *Proceedings of the IEEE* 86 (11) (1998) 2278–2324.
- [42] P. Chan, R. A. Curtis, J. Warwicker, Soluble expression of proteins correlates with a lack of positively-charged surface, *Scientific reports* 3 (2013) 3333.