International Journal of
Molecular Sciences

MDPI

Article

Protein–Protein Interactions Efficiently Modeled by Residue Cluster Classes

Albros Hermes Poot Velez [1], Fernando Fontove [2] and Gabriel Del Rio [1,*]

[1] Department of biochemistry and structural biology, Instituto de fisiologia celular, UNAM Mexico City 04510, Mexico; albros.poot@gmail.com
[2] C3 consensus, Leon Guanajuato 37266, Mexico; fernando.fontove@c3consensus.com
* Correspondence: gdelrio@ifc.unam.mx

Received: 19 April 2020; Accepted: 28 June 2020; Published: 6 July 2020

Abstract: Predicting protein–protein interactions (PPI) represents an important challenge in structural bioinformatics. Current computational methods display different degrees of accuracy when predicting these interactions. Different factors were proposed to help improve these predictions, including choosing the proper descriptors of proteins to represent these interactions, among others. In the current work, we provide a representative protein structure that is amenable to PPI classification using machine learning approaches, referred to as residue cluster classes. Through sampling and optimization, we identified the best algorithm–parameter pair to classify PPI from more than 360 different training sets. We tested these classifiers against PPI datasets that were not included in the training set but shared sequence similarity with proteins in the training set to reproduce the situation of most proteins sharing sequence similarity with others. We identified a model with almost no PPI error (96–99% of correctly classified instances) and showed that residue cluster classes of protein pairs displayed a distinct pattern between positive and negative protein interactions. Our results indicated that residue cluster classes are structural features relevant to model PPI and provide a novel tool to mathematically model the protein structure/function relationship.

Keywords: residue cluster class; protein–protein interaction; machine learning

1. Introduction

Proteins perform many vital functions in living organisms, with most depending on interactions with other molecules. Among these interactions, protein–protein interactions (PPI) are involved in maintaining cellular structure, regulating protein function, facilitating cellular transport, and ultimately encoding for the scaffold where most, if not all, cellular events take place [1–3]. Hence, identifying these PPI represent an important effort to characterize the molecular mechanisms at play in different living organisms.

To facilitate this effort, different computational approaches were described over the past years. Of particular interest to this work were those approaches based on machine learning (ML) techniques [4–6]. In ML, two global approaches are recognized, namely, supervised and unsupervised learning. In the latter, clusters of elements are identified and guided mainly according to the distance separating positive and negative PPI, while in supervised learning the algorithms thrive in identifying the frontiers of known clusters [7]. ML works on numerical representation of proteins; hence physical and chemical descriptors were developed to represent proteins [8–10] and used to predict PPI [11,12]. Other descriptors, such as proteins sequence composition [13,14], genomic data [15,16], and protein three-dimensional (3D) structures [17,18], among others [19,20], were described to represent proteins to predict PPI.

We previously reported a compact, 26-dimensional representation of protein structure based on residue cluster classes (RCCs) [21], which are obtained from the maximal cliques observed in the contact map derived from the protein 3D structure, taking into account the sequence localization of amino acid residues. RCCs are sets of maximal cliques that are grouped by size and classified according to the sequence proximity of the included residues (see Figure 1). RCCs represent the denser packing areas of a protein (every residue in these clusters is in contact with the rest). We showed that RCCs present a pattern that is recognizable by any heuristic ML approach and consequently provided a learnable representation for protein structure classification. Indeed, we showed that RCCs improved upon the state-of-the-art methods aimed to look for structural neighbors and structural classification [21]. In the present work, we aimed to test if protein structure is key to protein interactions through the use of RCCs to classify PPI.

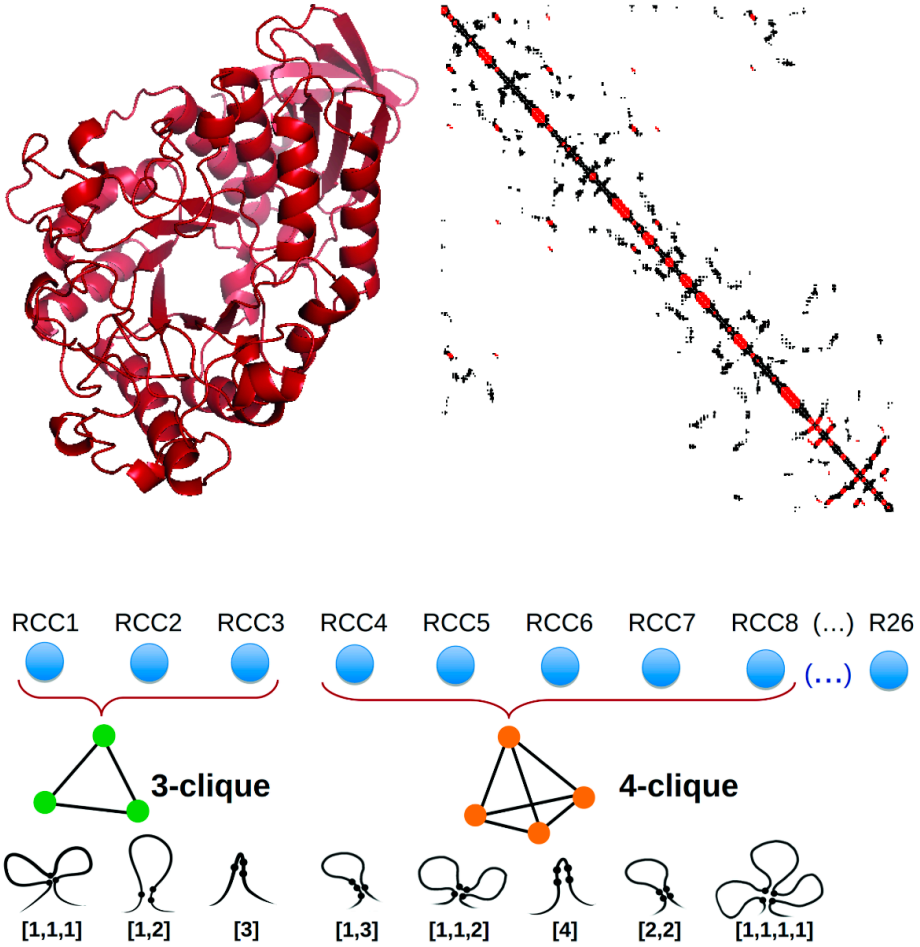

Figure 1. Residue cluster class (RCC) construction. Top panel: The atomic three-dimensional structure of a protein (left) is transformed into a contact map (right) that is used to build the RCC. Lower panel: The RCC is a 26-dimensional vector (RCC1, . . . , RCC26) derived by grouping residues close together in the three-dimensional space according to a distance criterion. These clusters group three (RCC1-RCC3), four (RCC4-RCC8), five (RCC9-RCC15), and six (RCC16-RCC26) amino acid residues; the image only represents RCC1–RCC8 for brevity. Different clusters of the same size are generated according to their sequence proximity. For instance, a cluster referred to as [1,1,1] represents three residues that are not proximal in the sequence (two or more residues apart) but are proximal to each other in the three-dimensional space.

2. Results

Experimentally determined positive examples of PPI were obtained from the three-dimensional interacting domains (3DID) database [22] and negative ones from the Negatome database [23] (see Methods); only proteins in these sets with 3D structures reported in the public repository of protein structures, Protein Data Bank (PDB), were included in this study. These sets rendered a database with 171,142 pairs of positive PPI and 692 pairs of proteins known to not interact (negative PPI) to use for training (see Methods), representing 99.6% and 0.4% of the training set, respectively. For every protein in these sets, we used 12 different representations of RCCs that were generated using 12 different distance criteria (see Methods). We also created two different sets of RCCs either including or not including the atoms of the sidechains of residues. Hence, every protein was represented in 24 different RCCs. Each of these representations was trained to model PPI either by adding every pair of RCCs or by producing the concatenation set (see Methods); in this way, every positive and negative PPI example was represented in 48 different forms (see Figure 2 and Methods). For every training set, we generated a complementary testing set that did not share the same positive PPI pair included in the training set, but included the same protein family (PFAM) domains (proteins in the same PFAM share ≥30% sequence identity) and negative PPI set (see Methods). The test set contained 4819 positive PPI instances (87.4%) and 692 negative ones (12.6%), hence, a naive predictor would predict 99.6% of all instances as positive PPI, rendering an error ≥12%. Thus, the testing set shared ≥30% sequence similarity with the training set, representing many cases of PPI. The files used for training and testing are available at https://github.com/gdelrioifc/PPI-RCC.

We first investigated whether positive and negative sets were separated by a simple distance criterion. To this end, we calculated the diameter D of the negative set of PPI (maximum Euclidian distance between any pair of RCCs within the same set) and compared this with the smallest distance d between the positive and negative sets. If positive and negative sets were separable by a distance criterion then, then $d < D$. Figure 3 shows the D and d values obtained for a subset (28) of the 48 training sets used in this work. We observed that these sets were not separable by a distance criterion.

Even though the sets were not separated by distance, a hyperplane was able to separate these two sets (see Methods) using RCCs built at 6 Å, hence, we concluded that these training sets were separable and consequently learnable (data not shown). To further explore these results, we observed that the distance values (d) for sets built at 3, 10, and 15 Å were equal to zero (see Figure 3), indicating that the positive and negative PPI were not separable at these distances used to build RCCs. The largest d values were observed at 6, 7, 8, and 9 Å, indicating that at such distances RCC representation of PPI facilitated the classification of positive from negative PPI. We provide a graphical interpretation regarding how the PPI space would look based on these results in Figure 4. Since we do not know the complete set of PPI, we expect that ML methods may fail to classify some PPI outside of the known regions for positive and negative. This is an intrinsic limitation of ML methods, but the geometrical representation by RCCs may provide ways to explore the limits of these positive and negative PPI regions, as discussed later.

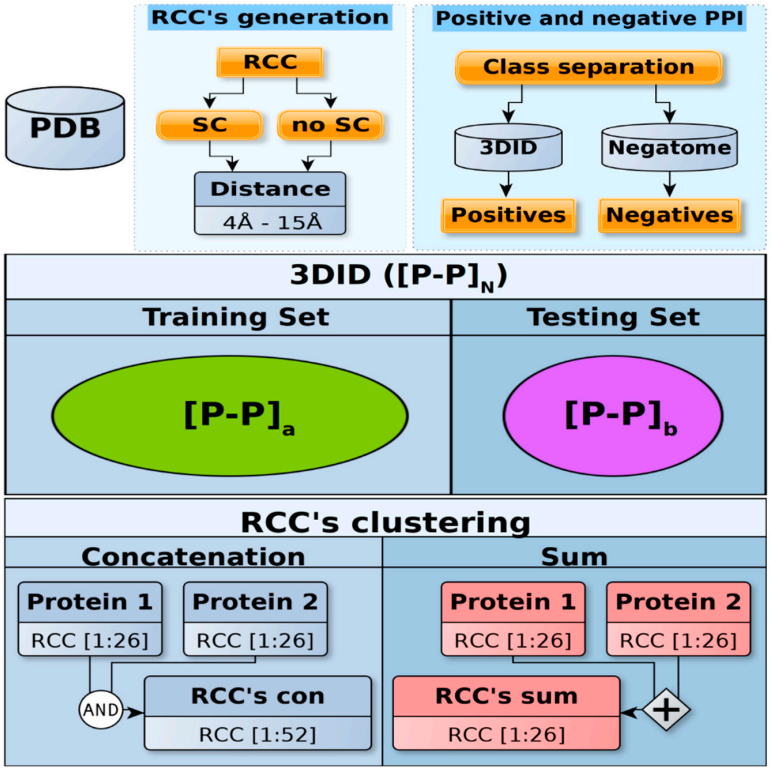

Figure 2. Dataset construction. Top panel: Every PDB reported as positive protein–protein interactions (PPI) in the 3DID database and as negative PPI in the Negatome database were processed to include (sidechain (SC)) or not include (no SC) the atoms of sidechains. A total of 12 different contact maps (contact distances 4–15 Å) were generated for each of these proteins and their residue cluster classes (RCCs) were calculated. Middle panel: The resulting numeric representations for all protein pairs were split into training and testing sets, which did not share the same positive PPI pair but did include the same PFAM domains. The procedure used to separate the positive PPI included in 3DID guaranteed that more instances were included in the training set than the testing set (see Methods). The positive PPI are represented in the figure as $[\text{P-P}]_{\text{N}}$ in the 3DID, $[\text{P-P}]_{\text{a}}$ for training, and $[\text{P-P}]_{\text{b}}$ for testing, where $N = a + b$ and $a > b$. Due to the limited number of negative cases, we used the same negative set for both training and testing sets. Lower panel: The sum (26 features) or concatenation (52 features) for each RCC in every pair of proteins was obtained. Finally, every PPI numeric representation was normalized, standardized, or kept in its original form (raw).

We searched for the best ML model and corresponding parameters for each of these representations through the optimization algorithm implemented in AutoWeka (see Methods). Our first approximation included all positive and all negative instances for all training sets. The motivation was to test if the RCCs could represent PPI with an unbalanced training set; not every unbalanced training set would render a biased classification. The test set included 4819 positive PPI that did not include the same pairs of PPI in the training set but shared sequence similarity. For this test, we used the same negative set of PPI for training and testing. To evaluate the quality of the models and any potential bias in their classification, we compared the percentage of correctly classified instances (% CCI) for every model derived from the training sets against the difference of % CCI in the training set versus the % CCI in the testing set (see Methods). We observed that building RCCs with sidechains at a distance of 6 Å rendered the best classification (100% of correctly classified instances in the testing set) either adding or concatenating individual RCCs of the participating proteins. The next best models were derived by the concatenation of

RCCs built at 5, 8, and 9 Å, including sidechains (see Supplementary Table S1 for models hyper parameters; the actual models can be found at https://github.com/gdelrioifc/PPI-RCC). It was noticeable that the best models performed a selection of features and no RCC position was discarded, indicating that all RCC values were relevant for the classification. Furthermore, the RCCs built at these distances were among those that separated the positive from the negative PPI (see Figure 3). For this set, the best algorithm was AdaBoost with J48 (a decision tree). AdaBoost is a metaclassifier that uses a set of classifiers (in this case several J48 classifiers) to obtain a new classifier that considers the weighted predictions of each individual classifier. These metaclassifiers are particularly useful when the border between classes is not easy to detect using an individual classifier. However, the second and third best algorithms, rendering 99.9% of correctly classified instances in the testing set, were the k-nearest neighbor and locally weighted learning (LWL) algorithms, indicating that the border was not difficult to identify. On the contrary, these results indicated that the border was learnable and many ML algorithms were able to identify it; indeed, 33 different models were able to classify positive from negative PPI, with ≥87.4% of correctly classified instances in the testing set. The nature of machine learning algorithms is to report a result even when it is not the best one. Hence, it is noteworthy that the best models rendered >95% of correctly classified instances.

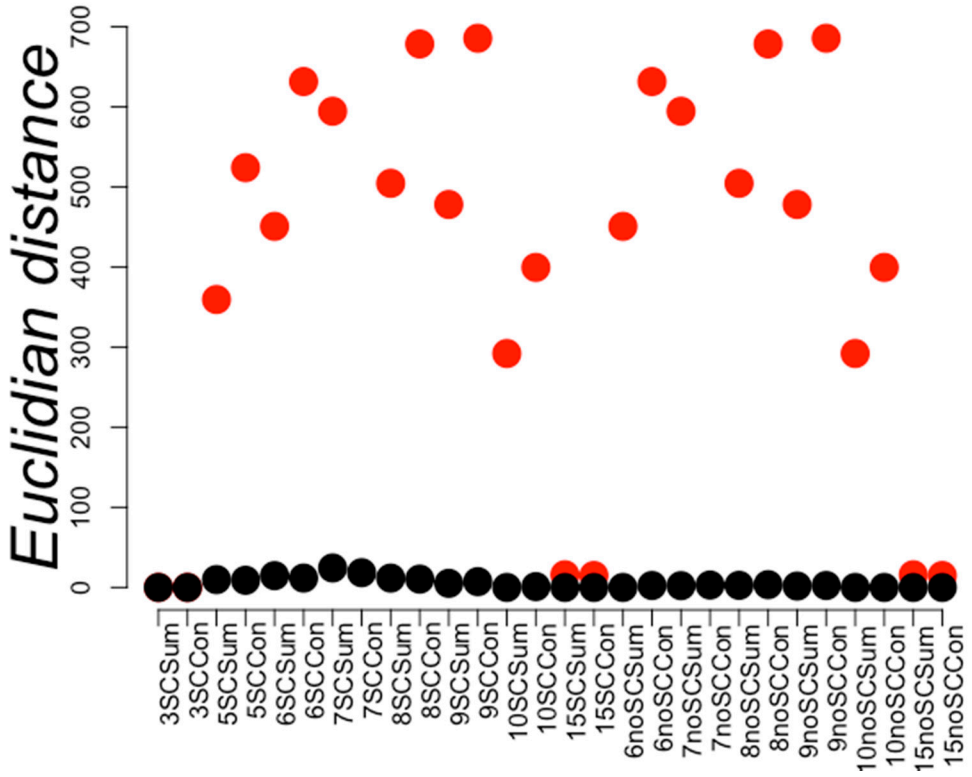

Figure 3. Distance within and between positive and negative PPI sets. Red circles represent the maximum Euclidian distances between instances of the same class (in this case, negative PPI) and the smallest distances between instances of different classes are shown in black circles. The *Y*-axis displays the distance values while the *X*-axis represents the different compared PPI representation compared. The first digit represents the distance used to build the RCCs and the next characters indicate whether the sidechains were included (SC) or not (noSC). Sum or Con indicates if the PPI was represented by the sum or concatenation of the individual RCC of the considered protein pair (see Methods). A red circle below the corresponding black circle indicates the separation of the positive PPI by distance from a negative PPI.

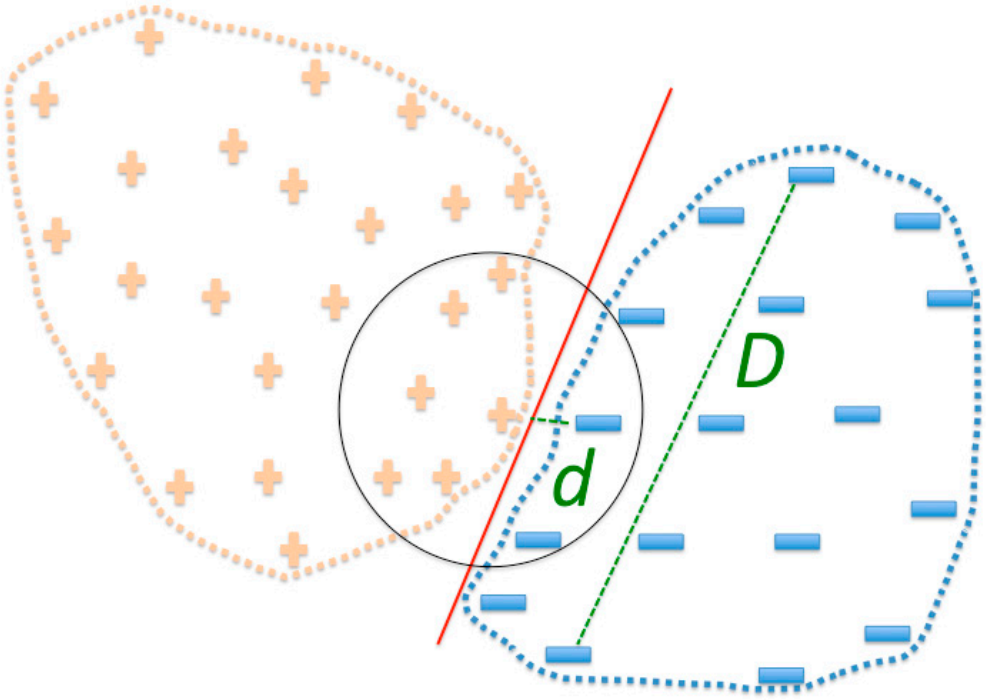

Figure 4. PPI space representation. Based on the distance separation of positive and negative PPI, we envisioned that positive PPI (plus symbols represent instances of positive PPI) would be dispersed over a large region in the space, but close to the negative PPI region(minus symbols represent instances of negative PPI); this proximity is likely the consequence of positive and negative PPI sharing sequence similarity. *D* represents the diameter of negative PPI, and the distance separating positive and negative PPI sets is represented by *d*. The red line separating positive and negative PPI sets represents a border that ML methods should be able to identify. The black circle represents k-nearest neighbors.

Since no RCC value was filtered out in the best models (those built at 5, 6, and 8 Å), we analyzed the frequency of RCC features present in any protein structure (i.e., how many times RCC1, RCC2, RCC3, . . . , RCC26 were found in all proteins). As shown in Figure 5, the RCCs built in the range from 7 to 8 Å with and without sidechains either adding or concatenating the RCCs from each protein had the largest proportion of nonzero values; very close to these were the RCCs built at 6 or 9 Å. It was noticeable that, of all the proteins analyzed, only RCCs built with 7 Å distance and without sidechains presented feature 16 (RCC16) with a value of zero value.

To evaluate if the positive and negative PPI displayed different distributions of RCC values, we performed a Wilcoxon test (See Methods). The comparison was performed for each of the 26 or 52 values for every protein pair included in the positive and negative PPI. As shown in Figure 6, all but one RCC value (RCC6) were significantly different in these two sets. This result was not consistent with the automatic selection of RCC features performed by AutoWeka, where no RCC coordinates/features were removed. It was expected that RCC6 and RCC16 would not be relevant for PPI classification, yet the performances of some of the trained models were perfect, suggesting that eliminating these two RCC positions may improve the classification in some of the worst models.

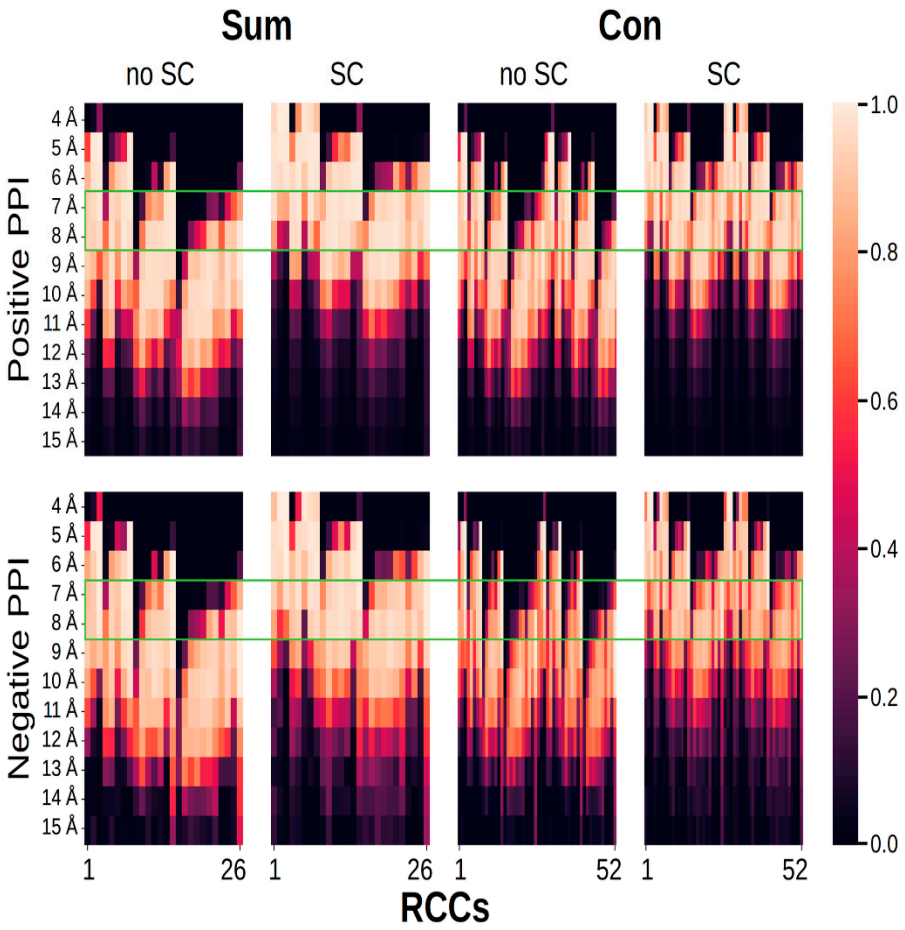

Figure 5. Fraction of RCCs with nonzero values. The graph shows eight different plots presenting the fraction of 26 (sum) or 52 (concatenation) values (*X*-axis) used to represent PPI (positive PPI) and pairs of proteins that did not interact (negative PPI); fractions are represented by a gradient color, with lighter colors representing those with more nonzero values. On the *Y*-axis, the distance used to build the contact map is presented (from 4 to 15 Å). The green rectangle indicates the set of distances used to build RCCs, where the number of features (RCC1, RCC2, RCC3, . . . , RCC26) with values equaling zero was minimum.

As noted above, the biased representation of positive examples of PPI in our trial may have induced a bias in the classification of the over-represented set (positive set). To further explore this idea, we performed a classification of PPI using diverse sampling procedures on the training sets. Each numeric representation was normalized, standardized. or kept according to its original values (see Figure 7 and Methods).

We used the conditions that rendered the lowest number of zero values on RCC positions. These included RCCs built using 7 and 8 Å; we also included the 6 Å for comparison with the conditions that rendered the best model using the full training set. For these RCCs, we scanned for the best model for all training sets via optimization performed by AutoWeka (see Methods). We generated 360 training sets that corresponded to the samplings performed to balance the training sets (see Methods). Each of the 360 models generated from the training sets were evaluated with 24 test sets (see Methods), rendering a total of 360 evaluation scores. To identify the best models, we plotted the percentages of correctly classified instances in every testing set against the differences of correctly classified instances between the testing and the training sets (see Figure 8), as we did before for the whole training set. The best five models were trained with RCCs without standardization or normalization, all using the locally weighted learning (LWL)

algorithm. These models rendered almost perfect predictions, with 99.6% of correctly classified instances, supporting the results previously observed using the whole training set. This indicated that the learning rate was not due to bias in the training set, but to the separation observed between the training and testing sets. Thus, avoiding the bias composition of positive instances in the training sets rendered models that classified correctly almost all instances in the testing set. In all these models, no RCC attributes were discarded during the optimization executed by AutoWeka. The hyperparameters for all 360 models are reported in Supplementary Table S2. The actual models can be found at https://github.com/gdelrioifc/PPI-RCC.

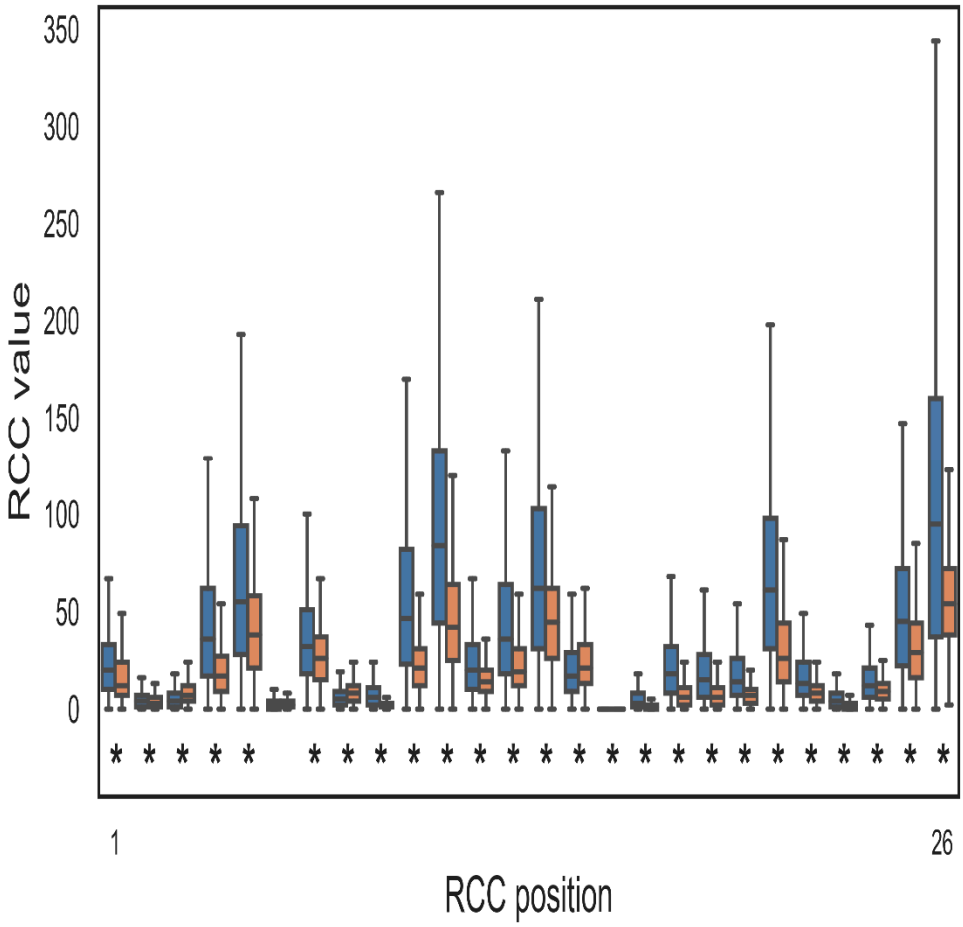

Figure 6. Comparison of positive and negative PPI through RCC. The image shows a representative example of the 16 performed comparisons (distance: 7, 8 Å; sidechains: yes, no; PPI RCC construction: sum, concatenation; statistical test: Wilcoxon–Bonferroni, Wilcoxon–Hochberg; all comparisons rendered very similar results (see https://github.com/gdelrioifc/PPI-RCC). The RCCs presented in the figure were obtained using a distance cutoff of 7 Å and included the residue sidechain atoms; the resulting RCCs for each protein pair were added. An asterisk is shown where the distribution of RCC values differs significantly between positive (blue) and negative (orange) PPI sets. The *X*-axis shows the RCC feature (RCC1, RCC2, . . . , RCC26) and the *Y*-axis represents the class of RCC and corresponding compared values. All but one RCC feature (RCC6) rendered a significantly different distribution of RCC values ($p < 0.5$; Wilcoxon test corrected by Bonferroni criterion).

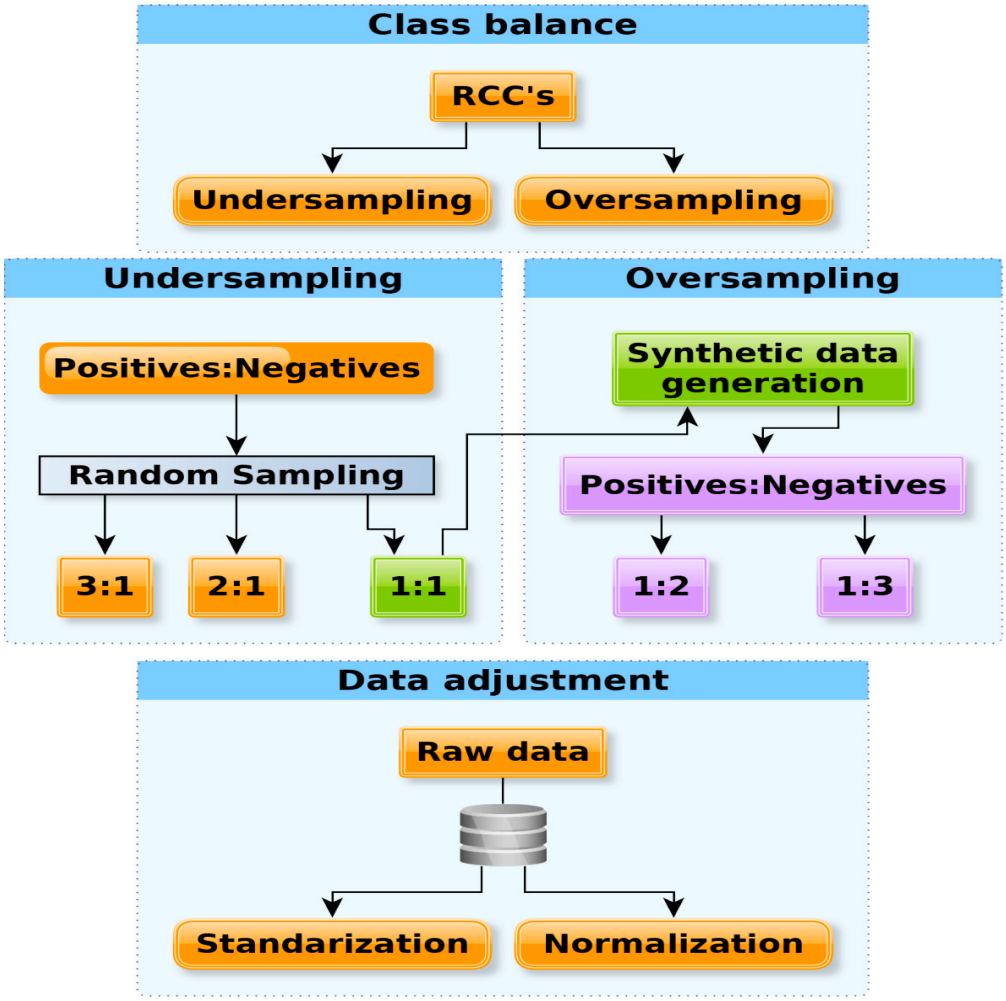

Figure 7. Sampling of training sets. Upper panel: Two different strategies were followed to deal with over-representation in the training sets, i.e., random elimination of instances from the over-represented positive PPI (under sampling) or generation of instances (synthetic sampling) from the under-represented negative PPI (over sampling). Middle panel: In under sampling, a random sample with equal proportions of positive and negative PPI (1:1), two times more positive than negative PPI (2:1), or three times more positive than negative PPI (3:1) were generated. In over sampling, the 1:1 sample generated in the under sampling, two times more negative than positive (1:2), and three times more negative than positive PPI (1:3) were generated. Lower panel: The RCCs generated for every PPI set were maintained (raw data), standardized, or normalized. The files used for training and testing are available at https://github.com/gdelrioifc/PPI-RCC.

The results of 6 Å did not improve the results obtained using 7 or 8 Å (data not shown). Comparing these results (99.6% of correctly classified instances) with those obtained with the full training set (100% of correctly classified instances in the testing set) indicated that the sampling did not reduce the positive and negative PPI regions. The fact that the LWL algorithm rendered the best predictions implied that the density of positive or negative PPI favored the corresponding class at any given point in the RCC space, even when the sampling produced the same number of positive and negative PPI or more negative than positive PPI; in other words, positive and negative PPI were separated in the RCC space (see Figure 4). LWL works in a similar way to k-nearest neighbors, but with a kernel (kernel regression); in this algorithm,

a test instance of PPI was classified according to its k-nearest neighbors from the training set and a weighted contribution of each neighbor [24].

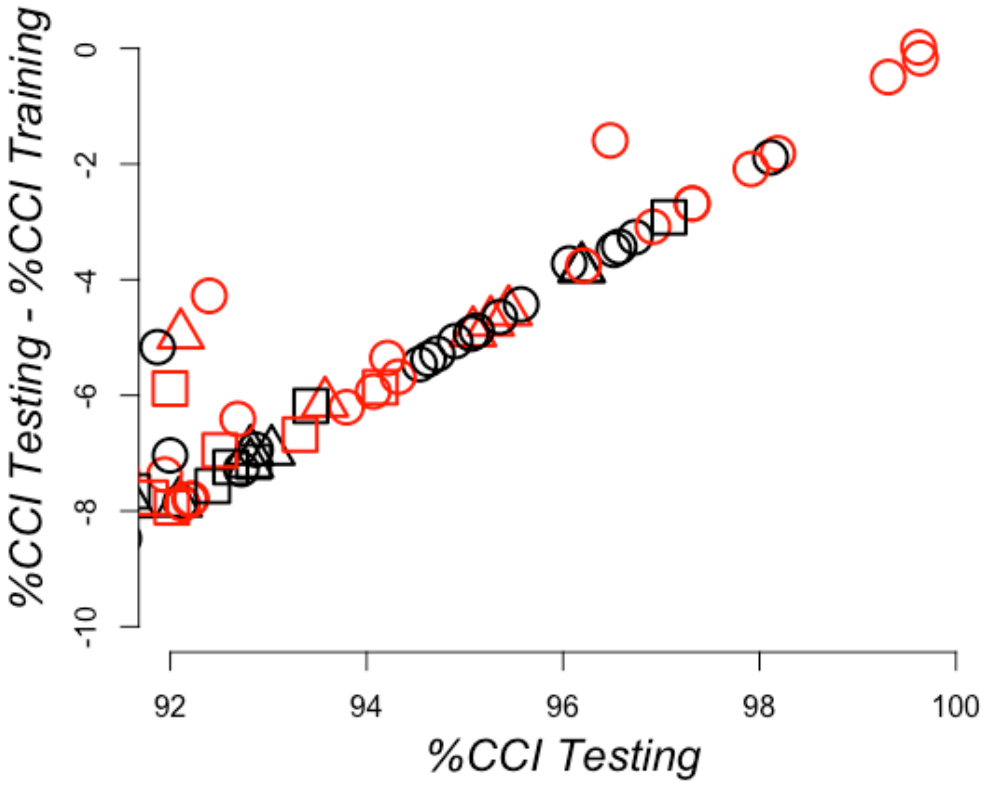

Figure 8. Learning efficiency on sampling training sets with redundancy. The percentages of correctly classified instances (CCI) for the testing sets (*X*-axis) are plotted against the differences observed in these values for the testing and training (*Y*-axes) sets. CCI corresponds to true predictions, including both positive and negative PPI. Therefore, 100% on the *X*-axes corresponds to not failing any prediction on the testing set and a 0 value on the *Y*-axis corresponds to no difference observed in the prediction between the training and the testing set. The best models are shown in the top right corner. Predictions achieved with raw RCCs are presented as circles, standardized RCCs as squares, and normalized RCCs as triangles. RCCs built using sidechains are otherwise shown in red and black.

Finally, we prepared a set of positive and negative instances of PPI without redundancy, that is, no RCCs used for training were found in the testing set and no RCCs in the training or testing sets were repeated (see Methods). This required reducing the number of instances. Both training and testing sets included 1:1, 2:1, and 3:1 sets (positives:negatives), as described in Figure 7 for RCCs built at 7 or 8 Å with or without sidechains. We searched for the best models using AutoWeka, as previously described (see Methods), and the results are summarized in Figure 9. The best model was identified used the LWL algorithm based on RCCs built at a distance of 7 Å with sidechains, achieving 96% CCI in the test set. The hyperparameters for these models are available in Supplementary Table S3. The actual training and test sets can be found at https://github.com/gdelrioifc/PPI-RCC.

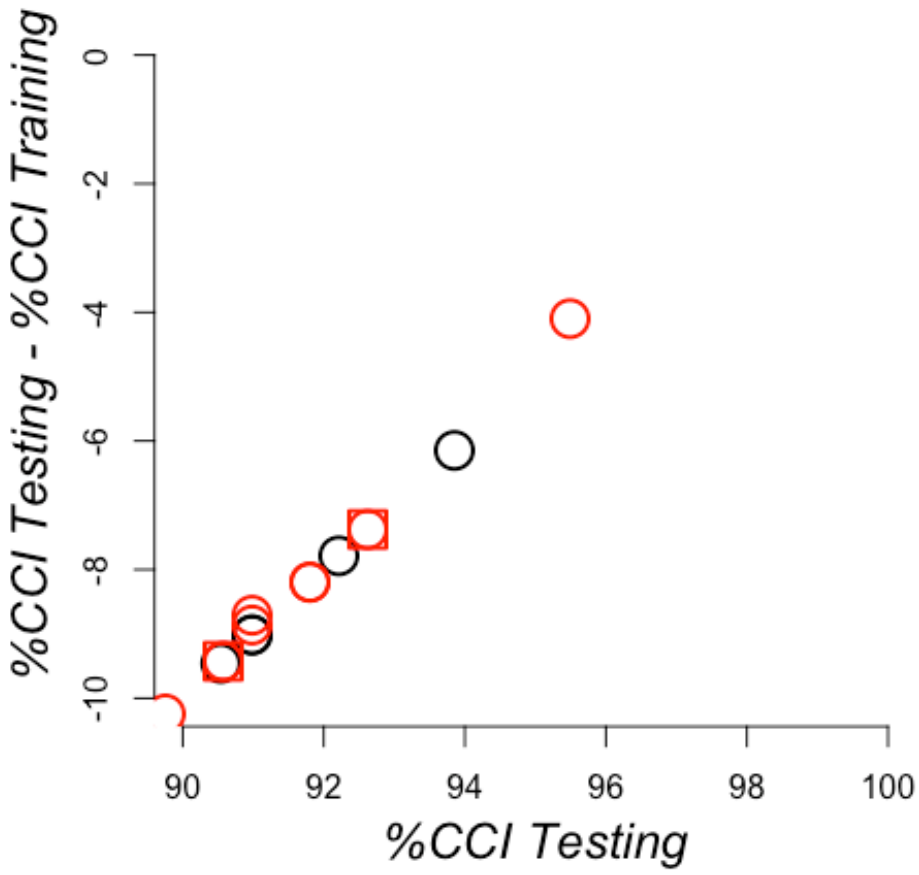

Figure 9. Learning efficiency on sampling training sets without redundancy. The percentage of correctly classified instances (CCI) for 360 testing sets (*X*-axis) is plotted against the difference observed in these values for testing and training (*Y*-axis). CCI corresponds to true predictions, including both positive and negative PPI. Therefore, 100% on the *X*-axes corresponds to not failing any prediction on the testing set and a 0 value on the *Y*-axis corresponds to no difference observed in the prediction between the training and the testing set. The best models are shown the top right corner. RCCs built using sidechains are otherwise represented by red circles or black squares.

3. Discussion

Our results indicated that the packing of proteins seemed to follow a distinct pattern that was more informative when the distance criterion used to create the contact map of residues was either 7 or 8 Å apart, either including or not including sidechains. We previously reported that these same distance cut-off values rendered the best models to classify protein structure (classification based on CATH database) and protein function (Gene Ontology annotations) from RCCs [25], presumably because, at these distances, the number of RCC values close to zero is minimal.

Another important aspect of our results was the efficiency achieved in the classification of PPI; either with the full set of positive and negative PPI or by different samplings, we achieved almost perfect classification and prediction. Our analysis regarding the distance of separation between positive and negative instances of PPI indicated that unsupervised learning may be efficient in classifying PPI, and our results provided evidence that supervised learning is very efficient in classifying positive and negative PPI. It was shown that the current methods to predict PPI displayed similar accuracy to high-throughput experimental bioassays [26,27] and, in the best case, prediction methods complement the experimental

results by prioritizing PPI for further experimental validation [28,29]. Thus, RCC performance is as good as or better than previous results, although this comparison should be observed with caution since different datasets were used.

For many PPI prediction methods, datasets were shown to be biased either because (i) the negative PPI dataset was not experimentally confirmed but assumed based on cellular localization [30], or (ii) the sequence similarity between training and testing sets was not considered, leading to overestimation of the prediction efficiency [31,32]. In the present work, we avoided these sources of bias in the construction of the training (Negatome database includes only experimentally validated negative PPI; see Methods) and testing datasets (no RCCs present in the training set were included in the test set). Our training and testing sets did share some degree of sequence similarity, but the separability between these sets indicated that we did not overestimate the learning rate of the best models. Considering that most proteins share some sequence similarity, we argue that our testing set is applicable for most cases. It would be relevant to evaluate RCC models to classify special cases of proteins with no sequence similarity, such as orphan proteins [33]. A more recent bias was noted involving proteins detectable by current experimental approaches, by cellular localization, or evolutionary lineage [34–36]; unfortunately, at this point we cannot address this bias and future studies should serve to correct or improve upon the results presented here.

We previously noted that the size (total number of amino acid residues) of proteins is linearly related to the number of clusters in the RCC [21], consistent with experimental observations that proteins present a near-constant density [37]. Based on our current results indicating that adding RCCs effectively represent PPI, it seems that protein–protein complexes may also linearly grow in number of clusters according to the number of residues (the sum of RCCs linearly increases the number of clusters with the molecular weight of the protein complex), or simply keep a constant density. In fact, it was reported that proteins and protein complexes show constant density as measured by electrospray ionization mass spectrometry and ion mobility [38]. Thus, our observation that the sum of individual RCCs serves to separate positive PPI from negative PPI may provide a geometrical approximation to study PPI that deserves to be further studied. In a similar fashion, the concatenation sets of RCCs projected two points in a 26-dimensional space into a single point of 52 dimensions; our results showed that the points in these 52 dimensions that represent positive PPI follow a distinct pattern different from those of negative PPI. This pattern may obey geometrical rules that could be studied using graph theory and/or algebra.

RCC requires knowledge of the 3D structure of the participating proteins, thereby limiting the applicability of this tool to proteins of known 3D structure. However, there are many methods to predict the 3D structure of proteins with different efficiencies [39,40]; it would be relevant to estimate how close a model should be to the real 3D structure to be useful in PPI predictions based on RCCs. The models reported here are not meant to quantify the strength of a PPI, but we anticipate this may be possible using RCCs. Our results indicated that the backbone conformation represented as RCCs contains enough information to model PPI, where the backbone conformation is the consequence of a particular set of sidechains, allowing RCCs to capture these details. Not including sidechain atoms is convenient to accelerate RCC computation, but does not make the prediction independent of the sidechains. These are the first results showing a clear relationship between protein 3D structure and PPI and highlight some intriguing possibilities that require future evaluation. For instance, it is possible that protein variants sharing very similar RCCs with the wild type keep the same interactions as the wild type sequence; in consequence, protein mutants significantly altering wild type RCCs should also alter protein interactions.

In summary, we provide the first evidence that PPI can be effectively modeled by combining individual RCCs of participating proteins. The use of RCCs provides a new perspective to geometrically study protein–protein interactions.

4. Materials and Methods

4.1. Datasets

Pairs of interacting proteins were obtained from the 3DID database [22] and the noninteracting protein pairs were derived from the Negatome database [23]. For every one of the proteins in these two sets, the corresponding three-dimensional structure was obtained from the Protein Data Bank [41]. The RCCs for each protein were calculated as previously described [21], but we varied the distance criterion from 4 to 15 Å (4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15 Å) and either included or did not include the atoms of the sidechains. Then, the resulting RCCs for every pair of proteins (positive and negative sets of PPI) were added or concatenated to produce a single numeric representation for every protein pair, with 26 or 52 features (RCC1, RCC2, RCC3, ... , RCC26, or RCC52). Finally, these sets were split into one set for training and one set for testing, with the training set including all the negative PPI. The testing set split the complete 3DID dataset to include a portion of the PFAM domains found in the 3DID dataset. Particularly, if a PFAM domain (PFAM1 for instance) was found N times in the 3DID database, only a portion of N was included in the testing set; the split was determined by the interacting proteins that contained the corresponding PFAM1 domain. In other words, if PFAM1 was found in the presence of PFAM2, and PFAM2 was found M times, and $N > M$, then PFAM1 would be found in the testing set M times. This way, we guaranteed that most PPI instances were included in the training set. The testing set included the complete Negatome, so training and testing sets had the same information on the negative PPI.

To study the effect of class imbalance present in the full training set, we generated a total of 360 training sets with two different sampling strategies (see below): 120 normalized sets (using Weka filter weka.filters.unsupervised.attribute.Normalize), 120 standardized sets (using Weka filter weka.filters.unsupervised.attribute.Standardize), and 120 raw sets. The 360 sets contained exactly the same 8 different combinations (7 or 8 Å; with or without sidechains; concatenation or sum of RCC), plus 3 undersamples (1:1, 2:1, 3:1) randomly generated 3 times (3×3), that were normalized, standardized, and kept as raw, rendering a subtotal of 216 ($8 \times 3 \times 3 \times 3$) training sets. Additionally, the 1:1 undersample was oversampled twice (1:2 and 1:3) and then normalized, standardized, and kept as raw, giving an additional subtotal of 144 ($8 \times 3 \times 2 \times 3$) sets. Thus, a total of 360 different training sets resulted. Two types of sampling procedures were employed from the Weka package [42], namely, undersampling and oversampling. Undersampling (supervised filter SpreadSubsample) used the larger set (positive PPI) to randomly select a sample whose size would be equal (1:1), twice (2:1), or triple (3:1) the size of the negative set. Oversampling used the Synthetic Minority Oversampling TEchnique (SMOTE) filter in Weka to generate double (1:2) or triple (1:3) instances of the negative set. All these oversamplings included one copy of the original negative set. The 1:1 sample generated by undersampling was used to generate the 1:2 and 1:3 oversampling sets. For testing, we generated 24 different sets for all 360 sets, derived from RCCs built using a contact distance of 7 or 8 Å, either using or not using the sidechains and adding or concatenating RCCs; they were then normalized, standardized, and kept as raw, rendering a total of 24 ($2 \times 2 \times 2 \times 3$). In summary, there were 8 testing sets for every 120 training sets for each data adjustment type (normalized, standardized, and raw). The numbers of positive and negative instances for training and testing in these sampling processes are indicated in Table 1.

A third experiment was conducted with datasets that were selected to avoid any repetition of RCCs within the same class (positive or negative) and between classes (positive and negative). In these datasets, the negative instances represented 80% in the training sets and the other 20% were posed in the testing sets. A total of 360 training sets were generated as previously described (216 training sets and 144 testing sets), including 1:1, 1:2, 1:3, 2:1, and 3:1 samplings (see Table 2).

Table 1. Test sets with redundancy.

P:N	Training		Testing	
	P	N	P	N
1:1	692	692	4819	692
2:1	1384	692	4819	692
3:1	2076	692	4819	692
1:2	692	1384	4819	692
1:3	692	2076	4819	692

P: Positive PPI; N: Negative PPI. Numbers in the table represent the number of instances for each dataset. For instance, 1:1 stands for samples with equal numbers of positive and negative instances of PPI, and 1:2 stands for samples with twice the number of negative PPI than positive PPI.

Table 2. Test sets without redundancy.

		Training		Testing	
		Positives	Negatives	Positives	Negatives
Concatenation	1:1	489	489	122	122
	2:1	978	489	122	122
	3:1	1467	489	122	122
Sum	1:1	448	448	111	111
	2:1	896	448	111	111
	3:1	1344	448	111	111

4.2. Machine Learning and Statistical Testing

The best algorithms and hyperparameters for each training set were identified through Bayesian optimization implemented in AutoWeka [43]. This Weka plugin searches for the optimum algorithm and hyperparameters given a specified time for the search; in our case, we used 1500 min. Shorter times (500 min) reproduced the same results for most cases, hence, we assumed that 1500 min identified the best models.

To test for the separability of positive and negative PPI used in the training set, we conducted a simple classification with a support vector machine using a linear kernel in Weka.

To determine if the positive and negative PPI used in this study were different, we performed a Wilcoxon test and corrected the significance of the compared RCC positions using the Bonferroni or Benjamini Hochberg approaches. This test and accompanying corrections were performed using the Python libraries Pandas [44], Scipy [45], Numpy [46], Statsmodels [47], Seaborn [48], and Matplotlib [49].

Supplementary Materials: The following are available online at http://www.mdpi.com/1422-0067/21/13/4787/s1, Table S1: Hyper parameters for best models obtained with AutoWeka using full training sets, Table S2: Hyper parameters for best models obtained with AutoWeka using samplings of training sets with redundancy, and Table S3: Hyper parameters for best models obtained with AutoWeka using samplings of training sets without redundancy. Supplementary Materials for the training sets, testing sets, and best corresponding models can be found at https://github.com/gdelrioifc/PPI-RCC.

Author Contributions: Conceptualization, G.D.R.; methodology, G.D.R., A.H.P.V., and F.F.; software, G.D.R., A.H.P.V., and F.F.; validation, G.D.R. and A.H.P.V.; formal analysis, G.D.R. and A.H.P.V.; resources, G.D.R.; data curation, G.D.R., A.H.P.V., and F.F.; writing—original draft preparation, G.D.R. and A.H.P.V.; writing—review and editing, G.D.R., A.H.P.V., and F.F.; supervision, G.D.R.; funding acquisition, G.D.R. All authors read and agreed to the published version of the manuscript.

Funding: This research was funded by CONACyT ((CB252316) and PAPIIT (IN208817)).

Acknowledgments: To Maria Teresa Lara Ortiz for her technical assistance in the development of this work. To Augusto César Poot Hernandez, head of the Unidad de Bioinformática y Manejo de la Información of the Instituto de Fisiología Celular, UNAM, for his technical support for the development of this project.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study, in the collection, analyses, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

References

1. Carbon, S.; Douglass, E.; Dunn, N.; Good, B.; Harris, N.L.; Lewis, S.E.; Mungall, C.J.; Basu, S.; Chisholm, R.L.; Dodson, R.J.; et al. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Res.* **2019**, *47*, D330–D338. [CrossRef]
2. Wang, J.P.; Liu, B.; Sun, Y.; Chiang, V.L.; Sederoff, R.R. Enzyme-enzyme interactions in monolignol biosynthesis. *Front Plant Sci.* **2019**, *9*, 1942. [CrossRef] [PubMed]
3. Freilich, R.; Arhar, T.; Abrams, J.L.; Gestwicki, J.E. Protein-Protein Interactions in the Molecular Chaperone Network. *Acc. Chem. Res.* **2018**, *51*, 940–949. [CrossRef]
4. Zahiri, J.; Emamjomeh, A.; Bagheri, S.; Ivazeh, A.; Mahdevar, G.; Sepasi Tehrani, H.; Mirzaie, M.; Fakheri, B.A.; Mohammad-Noori, M. Protein complex prediction: A survey. *Genomics* **2020**, *112*, 174–183. [CrossRef] [PubMed]
5. Liu, S.; Liu, C.; Deng, L. Machine learning approaches for protein-protein interaction hot spot prediction: Progress and comparative assessment. *Molecules* **2018**, *23*, 2535. [CrossRef]
6. Kotlyar, M.; Rossos, A.E.M.; Jurisica, I. Prediction of Protein-Protein Interactions. *Curr. Protoc. Bioinform.* **2017**, *60*, 8.2.1–8.2.14. [CrossRef]
7. Bzdok, D.; Krzywinski, M.; Altman, N. Points of significance: Machine learning: Supervised methods. *Nat. Methods* **2018**, *15*, 5–6. [CrossRef]
8. Ruiz-Blanco, Y.B.; Paz, W.; Green, J.; Marrero-Ponce, Y. ProtDCal: A program to compute general-purpose-numerical descriptors for sequences and 3D-structures of proteins. *BMC Bioinform.* **2015**, *16*, 162. Available online: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-015-0586-0 (accessed on 3 February 2020). [CrossRef]
9. Shen, H.B.; Chou, K.C. PseAAC: A flexible web server for generating various kinds of protein pseudo amino acid composition. *Anal. Biochem.* **2008**, *373*, 386–388. [CrossRef]
10. Li, Z.R.; Lin, H.H.; Han, L.Y.; Jiang, L.; Chen, X.; Chen, Y.Z. PROFEAT: A web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence. *Nucleic Acids Res.* **2006**, *34*, w32–w37. [CrossRef]
11. Sarkar, D.; Saha, S. Machine-learning techniques for the prediction of protein–protein interactions. *J. Biosci.* **2019**, *44*, 104. [CrossRef]
12. Romero-Molina, S.; Ruiz-Blanco, Y.B.; Green, J.R.; Sanchez-Garcia, E. ProtDCal-Suite: A web server for the numerical codification and functional analysis of proteins. *Protein Sci.* **2019**, *28*, 1734–1743. Available online: http://www.ncbi.nlm.nih.gov/pubmed/31271472 (accessed on 3 February 2020). [CrossRef] [PubMed]
13. Chen, M.; Ju, C.J.T.; Zhou, G.; Chen, X.; Zhang, T.; Chang, K.W.; Zaniolo, C.W.; Wang, W. Multifaceted Protein-Protein Interaction Prediction Based on Siamese Residual RCNN. *Bioinformatics* **2019**, *35*, i305–i314. [CrossRef] [PubMed]
14. Hu, L.; Chan, K.C.C. Extracting Coevolutionary Features from Protein Sequences for Predicting Protein-Protein Interactions. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2017**, *14*, 155–166. [CrossRef] [PubMed]
15. Szklarczyk, D.; Gable, A.L.; Lyon, D.; Junge, A.; Wyder, S.; Huerta-Cepas, J.; Simonovic, M.; Doncheva, N.T.; Morris, J.H.; Bork, P.; et al. STRING v11: Protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.* **2019**, *47*, D607–D613. [CrossRef]
16. Ardakani, F.B.; Schmidt, F.; Schulz, M.H. Predicting transcription factor binding using ensemble random forest models [version 2; peer review: 2 approved]. *F1000Research* **2019**, *7*, 1603. [CrossRef]

17. Hue, M.; Riffle, M.; Vert, J.P.; Noble, W.S. Large-scale prediction of protein-protein interactions from structures. *BMC Bioinform.* **2010**, *11*, 144. [CrossRef]
18. Chang, J.W.; Zhou, Y.Q.; Ul Qamar, M.T.; Chen, L.L.; Ding, Y.D. Prediction of protein–protein interactions by evidence combining methods. *Int. J. Mol. Sci.* **2016**, *17*, 1946. [CrossRef]
19. Ding, Z.; Kihara, D. Computational Methods for Predicting Protein-Protein Interactions Using Various Protein Features. *Curr. Protoc. Protein Sci.* **2018**, *93*, e62. Available online: http://doi.wiley.com/10.1002/cpps.62 (accessed on 3 February 2020). [CrossRef]
20. Zhang, S.B.; Tang, Q.R. Protein-protein interaction inference based on semantic similarity of Gene Ontology terms. *J. Theor. Biol.* **2016**, *401*, 30–37. [CrossRef]
21. Corral-Corral, R.; Chavez, E.; Del Rio, G. Machine Learnable Fold Space Representation based on Residue Cluster Classes. *Comput. Biol. Chem.* **2015**, *59*, 1–7. [CrossRef] [PubMed]
22. Mosca, R.; Céol, A.; Stein, A.; Olivella, R.; Aloy, P. 3did: A catalog of domain-based interactions of known three-dimensional structure. *Nucleic Acids Res.* **2014**, *42*, D374. [CrossRef] [PubMed]
23. Blohm, P.; Frishman, G.; Smialowski, P.; Goebels, F.; Wachinger, B.; Ruepp, A.; Frishman, D. Negatome 2.0: A database of non-interacting proteins derived by literature mining, manual annotation and protein structure analysis. *Nucleic Acids Res.* **2014**, *42*, D396–D400. Available online: http://www.ncbi.nlm.nih.gov/pubmed/24214996 (accessed on 16 February 2020). [CrossRef] [PubMed]
24. Atkeson, C.G.; Moore, A.W.; Schaal, S. Locally Weighted Learning. *Artif. Intell. Rev.* **1997**, *11*, 11–73. [CrossRef]
25. Fontove, F.; Del Rio, G. Residue cluster classes: A unified protein representation for efficient structural and functional classification. *Entropy* **2020**, *22*, 472. [CrossRef]
26. Zhang, Q.C.; Petrey, D.; Deng, L.; Qiang, L.; Shi, Y.; Thu, C.A.; Bisikirska, B.; Lefebvre, C.; Accili, D.; Hunter, T.; et al. Structure-based prediction of protein-protein interactions on a genome-wide scale. *Nature* **2012**, *490*, 556–560. [CrossRef]
27. Elefsinioti, A.; Saraç, Ö.S.; Hegele, A.; Plake, C.; Hubner, N.C.; Poser, I.; Sarov, M.; Hyman, A.; Mann, M.; Schroeder, M.; et al. Large-scale de novo prediction of physical protein-protein association. *Mol. Cell. Proteomics* **2011**, *10*, M111.010629. [CrossRef]
28. Petschnigg, J.; Groisman, B.; Kotlyar, M.; Taipale, M.; Zheng, Y.; Kurat, C.F.; Sayad, A.; Sierra, J.R.; Usaj, M.M.; Snider, J.; et al. The mammalian-membrane two-hybrid assay (MaMTH) for probing membrane-protein interactions in human cells. *Nat. Methods* **2014**, *11*, 585–592. [CrossRef]
29. Schwartz, A.S.; Yu, J.; Gardenour, K.R.; Finley, R.L.; Ideker, T. Cost-effective strategies for completing the interactome. *Nat. Methods* **2009**, *6*, 55–61. [CrossRef]
30. Ben-Hur, A.; Noble, W.S. Choosing negative examples for the prediction of protein-protein interactions. *BMC Bioinform.* **2006**, *7*, S2. [CrossRef]
31. Hamp, T.; Rost, B. More challenges for machine-learning protein interactions. *Bioinformatics* **2015**, *31*, 1521–1525. Available online: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu857 (accessed on 19 February 2020). [CrossRef] [PubMed]
32. Park, Y.; Marcotte, E.M. Flaws in evaluation schemes for pair-input computational predictions. *Nat. Methods* **2012**, *9*, 1134–1136. [CrossRef] [PubMed]
33. Basile, W.; Sachenkova, O.; Light, S.; Elofsson, A. High GC content causes orphan proteins to be intrinsically disordered. *PLoS Comput. Biol.* **2017**, *13*, e1005375. Available online: https://dx.plos.org/10.1371/journal.pcbi.1005375 (accessed on 1 April 2020). [CrossRef] [PubMed]
34. Kotlyar, M.; Pastrello, C.; Sheahan, N.; Jurisica, I. Integrated interactions database: Tissue-specific view of the human and model organism interactomes. *Nucleic Acids Res.* **2016**, *44*, D536–D541. Available online: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkv1115 (accessed on 19 February 2020). [CrossRef] [PubMed]
35. Snider, J.; Kotlyar, M.; Saraon, P.; Yao, Z.; Jurisica, I.; Stagljar, I. Fundamentals of protein interaction network mapping. *Mol. Syst. Biol.* **2015**, *11*, 848. Available online: https://onlinelibrary.wiley.com/doi/abs/10.15252/msb.20156351 (accessed on 19 February 2020). [CrossRef] [PubMed]

36. Wang, Z.; Clark, N.R.; Ma'ayan, A. Dynamics of the discovery process of protein-protein interactions from low content studies. *BMC Syst. Biol.* **2015**, *9*, 26. Available online: https://bmcsystbiol.biomedcentral.com/articles/10.1186/s12918-015-0173-z (accessed on 19 February 2020). [CrossRef] [PubMed]
37. Fischer, H.; Polikarpov, I.; Craievich, A.F. Average protein density is a molecular-weight-dependent function. *Protein Sci.* **2009**, *13*, 2825–2828. [CrossRef]
38. Kaddis, C.S.; Lomeli, S.H.; Yin, S.; Berhane, B.; Apostol, M.I.; Kickhoefer, V.A.; Rome, L.H.; Loo, J.A. Sizing Large Proteins and Protein Complexes by Electrospray Ionization Mass Spectrometry and Ion Mobility. *J. Am. Soc. Mass Spectrom.* **2007**, *18*, 1206–1216. [CrossRef]
39. Alquraishi, M.; Valencia, A. AlphaFold at CASP13. *Bioinformatics* **2019**, *35*, 4862–4865. Available online: http://www.ncbi.nlm.nih.gov/pubmed/31116374 (accessed on 19 February 2020). [CrossRef]
40. Roche, D.B.; McGuffin, L.J. Toolbox for protein structure prediction. *Methods in Molecular Biology* **2016**, *1369*, 363–377. [CrossRef]
41. Burley, S.K.; Berman, H.M.; Bhikadiya, C.; Bi, C.; Chen, L.; Di Costanzo, L.; Christie, C.; Dalenberg, K.; Duarte, J.M.; Dutta, S.; et al. RCSB Protein Data Bank: Biological macromolecular structures enabling research and education in fundamental biology, biomedicine, biotechnology and energy. *Nucleic Acids Res.* **2019**, *47*, D464–D474. Available online: https://academic.oup.com/nar/article/47/D1/D464/5144139 (accessed on 16 February 2020). [CrossRef]
42. Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I.H. The WEKA Data Mining Software: An Update. *ACM SIGKDD Explor. Newsl.* **2009**, *11*, 1. Available online: https://www.kdd.org/exploration_files/p2V11n1.pdf (accessed on 12 February 2020). [CrossRef]
43. Kotthoff, L.; Thornton, C.; Hoos, H.H.; Hutter, F.; Leyton-Brown, K. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *J. Mach. Learn. Res.* **2017**, *18*, 1–5. Available online: http://automl.org/autoweka (accessed on 12 February 2020).
44. Mckinney, W. Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference (SCIPY 2010), Austin, TX, USA, 28 June–3 July 2010; pp. 56–61.
45. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef] [PubMed]
46. Van Der Walt, S.; Colbert, S.C.; Varoquaux, G. The NumPy array: A structure for efficient numerical computation. *Comput. Sci. Eng.* **2011**, *13*, 22–30. [CrossRef]
47. Seabold, S.; Perktold, J. Statsmodels: Econometric and Statistical Modeling with Python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June–3 July 2010; pp. 92–96. Available online: http://statsmodels.sourceforge.net/ (accessed on 5 April 2020).
48. Waskom, M. Seaborn: Statistical Data Visualization—Seaborn 0.10.0 Documentation. 2012. Available online: https://seaborn.pydata.org/ (accessed on 9 April 2020).
49. Hunter, J.D. Matplotlib: A 2D graphics environment. *Comput. Sci. Eng.* **2007**, *9*, 99–104. [CrossRef]