OXFORD

*Briefings in Bioinformatics*, 2024, **25(2)**, bbae076

https://doi.org/10.1093/bib/bbae076

**Review**

# Cracking the black box of deep sequence-based protein–protein interaction prediction

Judith Bernett, David B. Blumenthal† and Markus List†

Corresponding author: Markus List, Data Science in Systems Biology, TUM School of Life Sciences, Technical University of Munich, Maximus-von-Imhof Forum 3, 85354, Freising, Germany. markus.list@tum.de

†Joint senior authors.

## Abstract

Identifying protein–protein interactions (PPIs) is crucial for deciphering biological pathways. Numerous prediction methods have been developed as cheap alternatives to biological experiments, reporting surprisingly high accuracy estimates. We systematically investigated how much reproducible deep learning models depend on data leakage, sequence similarities and node degree information, and compared them with basic machine learning models. We found that overlaps between training and test sets resulting from random splitting lead to strongly overestimated performances. In this setting, models learn solely from sequence similarities and node degrees. When data leakage is avoided by minimizing sequence similarities between training and test set, performances become random. Moreover, baseline models directly leveraging sequence similarity and network topology show good performances at a fraction of the computational cost. Thus, we advocate that any improvements should be reported relative to baseline methods in the future. Our findings suggest that predicting PPIs remains an unsolved task for proteins showing little sequence similarity to previously studied proteins, highlighting that further experimental research into the 'dark' protein interactome and better computational methods are needed.

***Keywords***: protein–protein Interaction Prediction; Data Leakage; Deep Learning

## INTRODUCTION

Proteins carry out essential biological functions, many of which require proteins to act jointly or to form complexes. Hence, identifying all pairwise interactions of proteins is an essential systems biology challenge toward understanding biological pathways and their dysregulation in diseases. Several technologies (e.g. yeast-2-hybrid screens, affinity purification mass-spectrometry) have been developed to unravel individual protein–protein interactions (PPIs), yielding large-scale PPI networks [1]. As it is not feasible to study all protein pairs exhaustively, a plethora of computational methods have been developed to predict PPIs as a binary classification task. Such methods often use only sequence information in various machine learning (ML) strategies, ranging from classical support vector machines (SVMs) to the most complex deep learning (DL) architectures currently conceivable [2–21]. These DL methods typically report phenomenal prediction accuracies in the range of 95–99%.

Though it was criticized that only a few of these methods have source code available and are reproducible [2, 22], it has not yet been examined systematically whether and how such results are possible. Since proteins interact in 3D space, predicting an interaction should implicitly consider the 3D structure of complexes, binding pockets, domains, surface residues and binding affinities. However, predicting protein 3D structure from sequence is an infamously hard problem area in which only recently AlphaFold2 had made a tremendous leap using a very complex model architecture and vast resources [23]. Moreover, predicting the structure of multi-chain complexes observed in PPIs remains an open challenge [24]. In light of this, the observed high accuracies for predicting PPIs from sequence information alone seem dubious.

Few studies shed light on the phenomenal accuracies reported for deep sequenced-based PPI prediction approaches: Almost all PPI datasets used for evaluating such approaches are randomly split into train and test sets using cross-validation. Park & Marcotte [25] showed that this causes an inflation of prediction performance due to training data leakage [25–27]. Upon random splitting, the same proteins occur both in the train and the test set, such that these sets are no longer independent [28]. For an extensive definition of data leakage, see [29, 30]. To quantify the effect of data leakage, Park & Marcotte [25] proposed three classes C1 (both proteins in a test pair occur in training), C2 (only one protein in a test pair occurs in training), and C3 (no overlap between training and test). Prediction accuracies usually drop significantly between C1 and C2 as well as between C2 and C3. It has also been shown that when datasets contain sequences with high pairwise sequence similarities, models overfit and accuracies are overestimated, giving a wrong impression of the state of the field [22, 26, 27]. Hamp & Rost [26], therefore, extended the requirements by demanding that, for C3, no test protein should be sequence-similar to a training protein (for C2 only one, for C1 both), and obtained similar results as Park & Marcotte.

**Judith Bernett** is a PhD candidate at the Technical University of Munich.
**David B. Blumenthal** is a professor and head of the Biomedical Network Science Lab at the Department Artificial Intelligence in Biomedical Engineering of the Friedrich-Alexander-Universität Erlangen-Nürnberg. He obtained his PhD in Computer Science from the Free University of Bozen-Bolzano.
**Markus List** is an assistant professor of Data Science in Systems Biology at the Technical University of Munich. He obtained his PhD at the University of Southern Denmark and was a postdoctoral fellow at the Max Planck Institute for Informatics.

**Received:** November 23, 2023. **Revised:** January 09, 2024

Furthermore, Chatterjee *et al.* [31] have recently shown that DL methods for protein–ligand prediction use degree information as shortcuts instead of learning from sequence features. A baseline model using only topology information performs equally well for that task. They reveal that protein hubs have disproportionally more positive annotations and that most proteins and ligands either have almost no positive or almost no negative annotations, which the methods leverage.

In addition to positive examples, it is crucial to add sufficiently realistic negative examples to both train and test sets. These can be randomly sampled by choosing protein pairs not reported as PPIs in public databases. To avoid using false negatives, it is common practice to choose proteins that are not annotated to the same cellular compartment and are thus not expected to interact in a cell. However, Ben-Hur & Noble [32] have shown that this approach makes the learning problem considerably easier than it is in reality. Moreover, protein pairs are tested for interaction in an artificial system [33] and databases thus frequently include interactions of pairs annotated to different cellular locations.

In this work, we systematically examine three scenarios that might explain why sequence-based models correctly predict whether two proteins $p_1$ and $p_2$ interact:

- Explanation 1: The models can detect patterns in the sequences of $p_1$ and $p_2$ that are responsible for whether they can interact (e.g. matching binding sites, domains, motifs).
- Explanation 2: The models utilize node degree information shortcuts that individually explain whether the protein interacts. Based on these individual tendencies, they predict the interactions (e.g. if, in the training fold, $p_1$ only appears in interactions and never in the negative set, protein pairs from the training fold that involve $p_1$ will likely be predicted as interacting).
- Explanation 3: The models merely check whether $p_1$ and $p_2$ are similar to protein sequences $p'_1$ and $p'_2$ from the training set and make a prediction based on the interactions of $p'_1$ and $p'_2$ (e.g. if $p'_1$ interacts with $p'_2$, $p_1$ probably interacts with $p_2$ as well).

While, for many methods, the authors (implicitly) assume that their method's excellent prediction performance can be attributed to Explanation 1, we hypothesize that Explanation 2 and Explanation 3 are the actual drivers of high prediction accuracy. To investigate Explanation 2, we randomized the positive input network in the training data (but not in the test data) via degree-preserving rewiring. Hence, each protein's node degree is preserved, but the edges are no longer biologically plausible. If node degree information shortcuts indeed drove prediction performance, we would expect only a moderate drop in the test accuracy. Additionally, we incorporated two baseline methods (harmonic function [34] and local and global consistency algorithm [35]) that exclusively utilize network topology to infer if two proteins interact.

To investigate Explanation 3, we carried out a 2-fold strategy: On the one hand, we compared deep sequenced-based PPI prediction approaches against SPRINT [36]—an algorithmic PPI prediction approach based on local sequence alignment—as well as against simple baseline ML models which, by design, have access only to sequence similarity information. If sequence similarity indeed were a main driver of prediction performance, we would expect our baselines to achieve similar performances as the state-of-the-art DL methods. On the other hand, we partitioned the proteome into two blocks such that inter-block sequence similarities are minimized and selected PPIs for train and test sets from different blocks of the partition. This ensures that sequence similarity patterns learned during training cannot be leveraged at test time. If Explanation 3 were valid, we would expect a significant drop in prediction performance.

We conducted our analyses for six deep sequence-based PPI prediction methods [2, 4, 13, 20, 21], which we trained and tested on the same seven publicly available and commonly used datasets (two datasets with yeast PPIs [12, 37] and five with human PPIs [2, 11, 20, 38]). Our results show that training data leakage can fully explain the excellent accuracies reported in the literature. More specifically, if pairwise sequence similarities are minimized between disjoint training and test sets, performance is random, proving that sequence similarity and node degree are the only relevant features in current sequence-based PPI prediction methods. Finally, we generated a gold standard dataset to enable data-leakage-free validation of future PPI prediction methods.

# RESULTS

## Overview

We reviewed the literature for PPI prediction methods and their underlying datasets (Supplemental Table S1). For most of the 32 methods we found, extraordinary prediction performances are reported. However, source code is available only for 12 of them, emphasizing the reproducibility crisis in ML-based science [29]. Since we focused on understanding how sequence information contributes to DL-based PPI prediction, we selected methods that we managed to reproduce with reasonable effort and which rely exclusively on sequence information. This reduced the number of DL methods to Richoux-FC, Richoux-LSTM [2], DeepFE [13], PIPR [4], D-SCRIPT [20] and Topsy-Turvy [21].

For testing how much can be predicted from topology alone, we incorporated two node classification algorithms (harmonic function [34], local and global consistency [35]), which operate on the line graphs of the input networks. Additionally, we tested SPRINT [36], a fast algorithmic method that uses only sequence similarities of protein pairs to predict PPIs. We also included two baseline ML models (Random Forest, SVM) that used dimensionality-reduced (Principal Component Analysis (PCA), Multidimensional Scaling (MDS), node2vec) sequence similarity vectors as input for each protein. These baseline methods allowed us to assess the benefit of DL and to test the hypothesis that sequence similarity alone is already sufficient to achieve good prediction performance. Pairwise sequence similarities were precomputed by SIMAP2 [39]. Although sequence similarities were the only input feature, we note that the methods could learn node degrees implicitly during training. Supplemental Figure S1 depicts a schematic overview of the methods' principles.

We tested the methods on popular yeast and human datasets, the dataset used to validate the D-SCRIPT method (D-SCRIPT UNBALANCED) [20], and the two datasets by Richoux *et al.* [2] (see Table 1 for an overview). The latter two datasets were included because of their size and the unique generation of the strict test dataset, which was designed to be free from hub biases. D-SCRIPT UNBALANCED was included because it is deliberately unbalanced (1 to 10 positive versus negative annotations) to better reflect the underlying label distribution. The two Richoux datasets were created from a larger dataset consisting of PPIs annotated in Uniprot, which we later used for the partitioning task and refer to as RICHOUX-UNIPROT. All datasets were cleaned from duplicates and balanced except for D-SCRIPT UNBALANCED). Because of GPU restrictions, we created length-restricted datasets for D-SCRIPT and Topsy-Turvy, in which each protein had between 50 and 1000 amino acids. The original datasets were split 80/20 into

**Table 1:** Overview of datasets. $n$ denotes the overall number of samples in the datasets (after balancing and removal of duplicates), i.e. the number of PPIs plus the number of randomly sampled non-edges. $n_{\mathrm{restr.}}$ is size of the length-restricted datasets where both proteins of each (non-)interaction have between 50 and 1000 amino acids. These datasets were used for D-SCRIPT and Topsy-Turvy. $n_{\mathrm{method}}$ is number of methods found in our literature review that were tested on the respective datasets. MRA denotes the median reported accuracy of the $n_{\mathrm{method}}$ methods tested on the respective datasets. D-SCRIPT and Topsy-Turvy only reported auPR and AUC on their dataset.

| Dataset | Organism | $n$ | $n_{\mathrm{restr.}}$ | $n_{\mathrm{method}}$ | MRA (in %) |
|---|---|---|---|---|---|
| HUANG[40] | Human | 6690 | 4758 | 4 | 98.43 |
| GUO[37] | Yeast | 11 162 | 8760 | 14 | 94.75 |
| DU[12] | Yeast | 34 512 | 27 356 | 5 | 92.50 |
| PAN[38] | Human | 62 962 | 44 920 | 5 | 96.82 |
| RICHOUX-REGULAR[2] | Human | 79 868 | 67 724 | 2 | 88.10 |
| RICHOUX-STRICT[2] | Human | 68 664 | 78 776 | 2 | 77.29 |
| D-SCRIPT UNBALANCED | Human | 42 6492 | 426 283 | 2 | 0.5605 (auPR) |

training/test except for RICHOUX-REGULAR, RICHOUX-STRICT and D-SCRIPT UNBALANCED, which were already split into training/(validation/) test. Since we only used default hyperparameters, we did not need a validation set and added the validation to training for these two datasets. We chose the random 80/20 split since most reviewed methods report the mean accuracy of 5-fold cross-validation [4–8, 11, 18, 38, 41, 42] or a random holdout test set [2, 3, 12, 13, 37, 43–46]. To confirm that the composition of the resulting training and test set does not significantly impact the results, we split the original and rewired GUO and HUANG datasets 10 times with different seeds (see Supplementary Figures S30, S31, and Supplementary Tables S8, S9). As many datasets are rather small, those models which were developed for larger datasets (e.g. D-SCRIPT and Topsy-Turvy) could be prone to overfitting. We therefore also tested how early stopping influences the results of all DL methods. For picking the best model from the epochs, we randomly took 10% of the training set as validation set in this setting.

Figure 1 provides an overview of our analyses. We first consider a random split into train and test set, which we expect to introduce data leakage (see Methods for details). To test how much the models learn from node degree only (Explanation 2), we next rewired the positive PPIs (edges in PPI networks) in all training folds. For this, we randomly re-assigned all edges but preserved the expected node degree for each protein, rendering the new positive PPI networks biologically meaningless. Finally, we used the KaHIP [47] method with length-normalized, pre-computed SIMAP2 [39] bitscores as input to partition the human and yeast proteomes into two blocks, $P_0$ and $P_1$, such that pairs of protein sequences from different blocks are dissimilar. Then, for each dataset, all PPIs $(p_1, p_2)$ were assigned to three blocks $\boldsymbol{INTRA}_0$, $\boldsymbol{INTRA}_1$ and $\boldsymbol{INTER}$, depending, respectively, on whether $p_1$ and $p_2$ are both contained in $P_0$, whether they are both contained in $P_1$ or whether they are contained in different blocks of the partition $\{P_0, P_1\}$. If Explanations 2 and 3 apply, we expect a significant drop in accuracy if we train on the PPIs contained in $\boldsymbol{INTRA}_0$ and test on the ones contained in $\boldsymbol{INTRA}_1$, since there is no direct data leakage $(P_0 \cap P_1 = \emptyset)$ and a minimized amount of indirect data leakage due to sequence similarity. If we train on $\boldsymbol{INTER}$ and test on $\boldsymbol{INTRA}_0$ or $\boldsymbol{INTRA}_1$, we expect a smaller drop in accuracy since there is data leakage.

## Results on randomly split original benchmark datasets

Figure 2 shows our results upon randomly splitting the original benchmark datasets into 80% train / 20% test. Since all published methods except for D-SCRIPT and Topsy-Turvy were reported to show close to perfect performances, we expected roughly comparable results across methods within each dataset. As larger data sets are more prone to data leakage, we further expected accuracy to increase with dataset size for random splitting.

Comparing the results for all datasets except for the RICHOUX-STRICT dataset, we can see that the area under the precision-recall curve (AUPR) values of SPRINT rise with the number of unique proteins in the dataset (Supplemental Table S2). SPRINT's AUPRs match the balanced accuracies of the DL models on large datasets, which shows that finding similar subsequences to predict PPIs is already sufficient to reach excellent performance measures when proteins between the train and test set are shared.

For almost all methods, performances are exceptionally high on the HUANG and PAN datasets. This phenomenon can be explained by looking at the node degree distributions of the positive and negative datasets (Figure 2(b)). While both distributions follow the power law for the other datasets (Supplemental Figure S3), the negative examples were sampled uniformly for the HUANG and PAN datasets. Methods can thus primarily distinguish between positive and negative examples by node degree alone.

We further secure this finding by closer inspection of the degree ratios. Supplemental Figure S6a shows that most proteins in HUANG and PAN have either exclusively positive or negative interactions annotated (degree ratios are mainly 1 or 0). Because of the substantial data leakage (Figure 2(c), Supplemental Table S2), the proportion of training proteins with a high or low degree ratio in the positive or negative parts of the test sets is very high for HUANG (87% and 86%) and PAN (92% and 90%, Supplemental Figure S7a). Suppose an algorithm correctly predicts all of these interactions because of the degree information shortcut and assigns a random label for the remaining test interactions. In that case, we expect an accuracy of 93.25% and 95.5%, respectively. This estimate is close to the prediction of most methods, including but not limited to the topology methods.

D-SCRIPT and Topsy-Turvy perform poorly due to overfitting (Supplemental Figures S8, S9). While training loss decreases and training accuracy increases, validation loss stays constant or increases, and validation accuracy decreases for all datasets. The only datasets where some learning is visible are HUANG and GUO, which is also reflected by the final reported balanced accuracy. While D-SCRIPT's final prediction performance on the PAN dataset is far above random, inspecting the loss and accuracy patterns over the 10 epochs reveals an overfitting pattern.

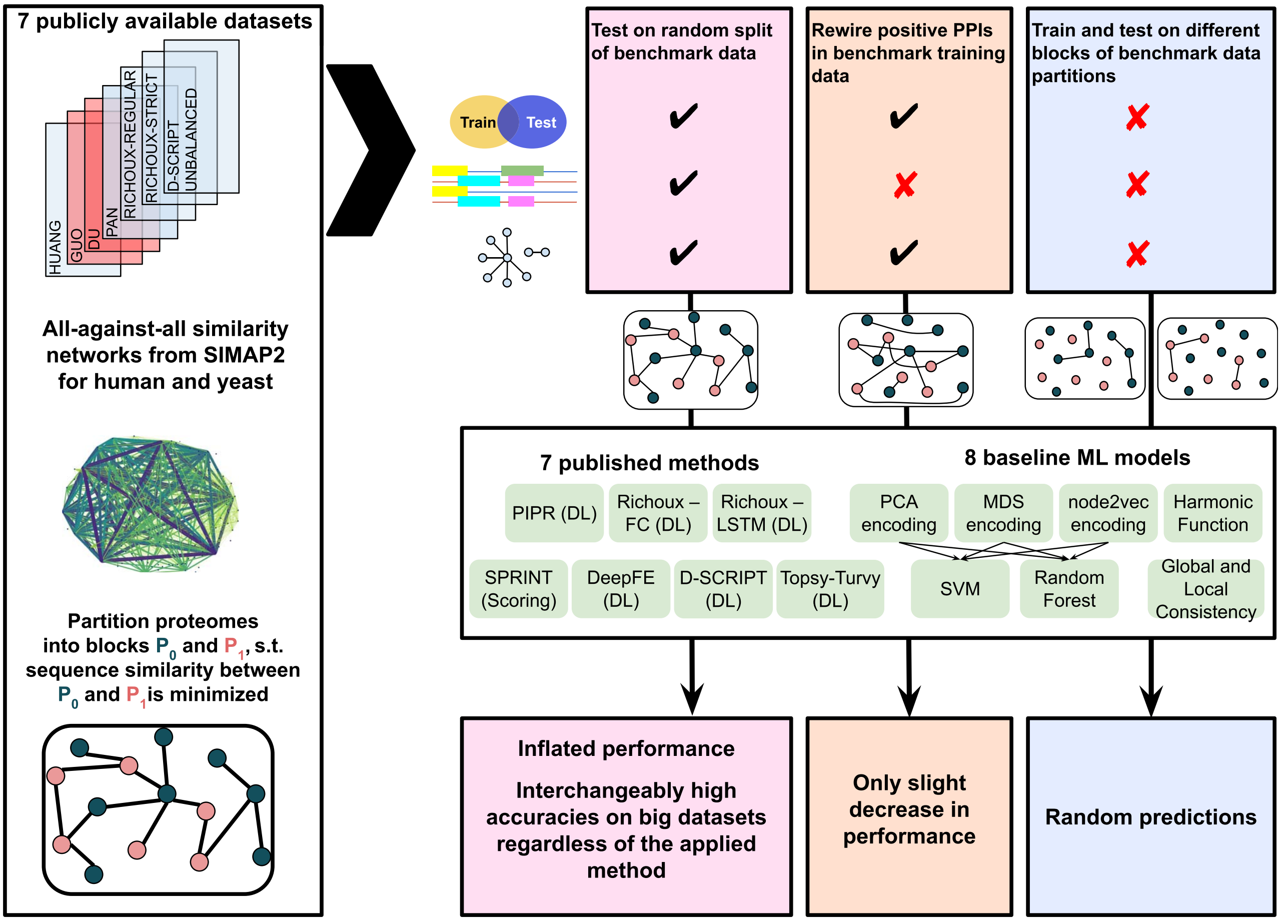
7 publicly available datasets
HUANG
GUO
DU
PAN
RICHOUX-REGULAR
RICHOUX-STRICT
D-SCRIPT
UNBALANCED
All-against-all similarity networks from SIMAP2 for human and yeast
Partition proteomes into blocks P0 and P1, s.t. sequence similarity between P0 and P1 is minimized
Train
Test
Test on random split of benchmark data
✓
✓
✓
Rewire positive PPIs in benchmark training data
✓
✗
✓
Train and test on different blocks of benchmark data partitions
✗
✗
✗
7 published methods
PIPR (DL)
Richoux – FC (DL)
Richoux – LSTM (DL)
SPRINT (Scoring)
DeepFE (DL)
D-SCRIPT (DL)
Topsy-Turvy (DL)
8 baseline ML models
PCA encoding
MDS encoding
node2vec encoding
Harmonic Function
SVM
Random Forest
Global and Local Consistency
Inflated performance
Interchangeably high accuracies on big datasets regardless of the applied method
Only slight decrease in performance
Random predictions

**Figure 1.** Overview of analyses. Seven publicly available datasets were used for testing seven published methods and eight basic ML models that use sequence similarities or topology as only input. We performed three tests to explain the phenomenal accuracies reported for DL methods: (1) We split the original data randomly into 80% train / 20% test, introducing data leakage through overlap of train and test proteins. Methods could learn from node degree biases and sequence similarities. This yielded inflated performance estimates and interchangeably high accuracies for DL and basic models on large enough datasets. (2) We rewired the positive train PPIs such that models could only learn from node degrees. Nevertheless, performance estimates only decreased slightly. (3) We partitioned the human and yeast proteomes into two blocks $P_0$ and $P_1$ such that proteins from different blocks have pairwise dissimilar sequences and assigned PPIs $(p_1, p_2)$ to blocks $\mathit{INTRA}_0$, $\mathit{INTRA}_1$ and $\mathit{INTER}$, depending on whether $p_1$ and $p_2$ are both contained in $P_0$ or $P_1$, or fall into different blocks of the partition. When trained on $\mathit{INTRA}_0$ and tested on $\mathit{INTRA}_1$ (no overlap between train and test data, models could neither learn from sequence similarity nor from node degrees), all tested models predicted PPIs randomly.

Both models mostly predict test candidates to be non-interacting (specificity ≈ 1.0, see Supplemental Figure S20).

Early stopping leads to a strong improvement of D-SCRIPT and Topsy-Turvy on almost all datasets (Supplemental Figure S15). The other methods, however, mostly lose performance. Many models already reach their best performances on the validation dataset in early epochs (Supplemental Figures S8, S9), indicating that the respective datasets might not be suitable for learning as they lead to immediate overfitting.

Except for SVM-based methods, the performance of the baseline ML methods is virtually interchangeable and roughly equal to the DL methods on the larger datasets, excluding D-SCRIPT and Topsy-Turvy. The random forest-based models seem to be a powerful alternative to the DL models.

As expected, the performance of all methods drops significantly on RICHOUX-STRICT. As shown in Figure 2(c), all datasets except for RICHOUX-STRICT include the vast majority of the proteins in both the train and test set. Consequently, RICHOUX-STRICT is less prone to training data leakage, explaining the observed results. In the presence of data leakage, robust predictions can be made based on node degree and sequence similarity, even with basic ML models. RICHOUX-STRICT's overlap is still almost 50%, but it is free from hub biases. However, 40% of the positive and 51% of the negative test interactions still involve a protein with mainly positive or mainly negative annotations in the training set, respectively (Supplemental Figure S7). Applying the same logic as above, we expect an accuracy of 72.75%, which is the performance of most methods. SPRINT does not use node degrees for its predictions, so the number of protein pairs seen in training seems to be large enough for SPRINT to find similar subsequences for the RICHOUX-STRICT test set.

Due to memory restrictions, the harmonic function algorithm could not be run on the D-SCRIPT UNBALANCED dataset. SPRINT, both Richoux models, DeepFE, PIPR and the random forest-based methods could handle the 1 to 10 imbalance, while the other models performed close to random (balanced accuracy around 0.5). DeepFE and PIPR held up the good performance under early stopping and D-SCRIPT and Topsy-Turvy profited strongly, showing that when overfitting is prevented and the training set is large enough, models can generalize to the test set.

For all benchmark datasets, however, the overall number of proteins is far from the real number of proteins (Supplemental Table S2). It can hence be expected that the models overfit

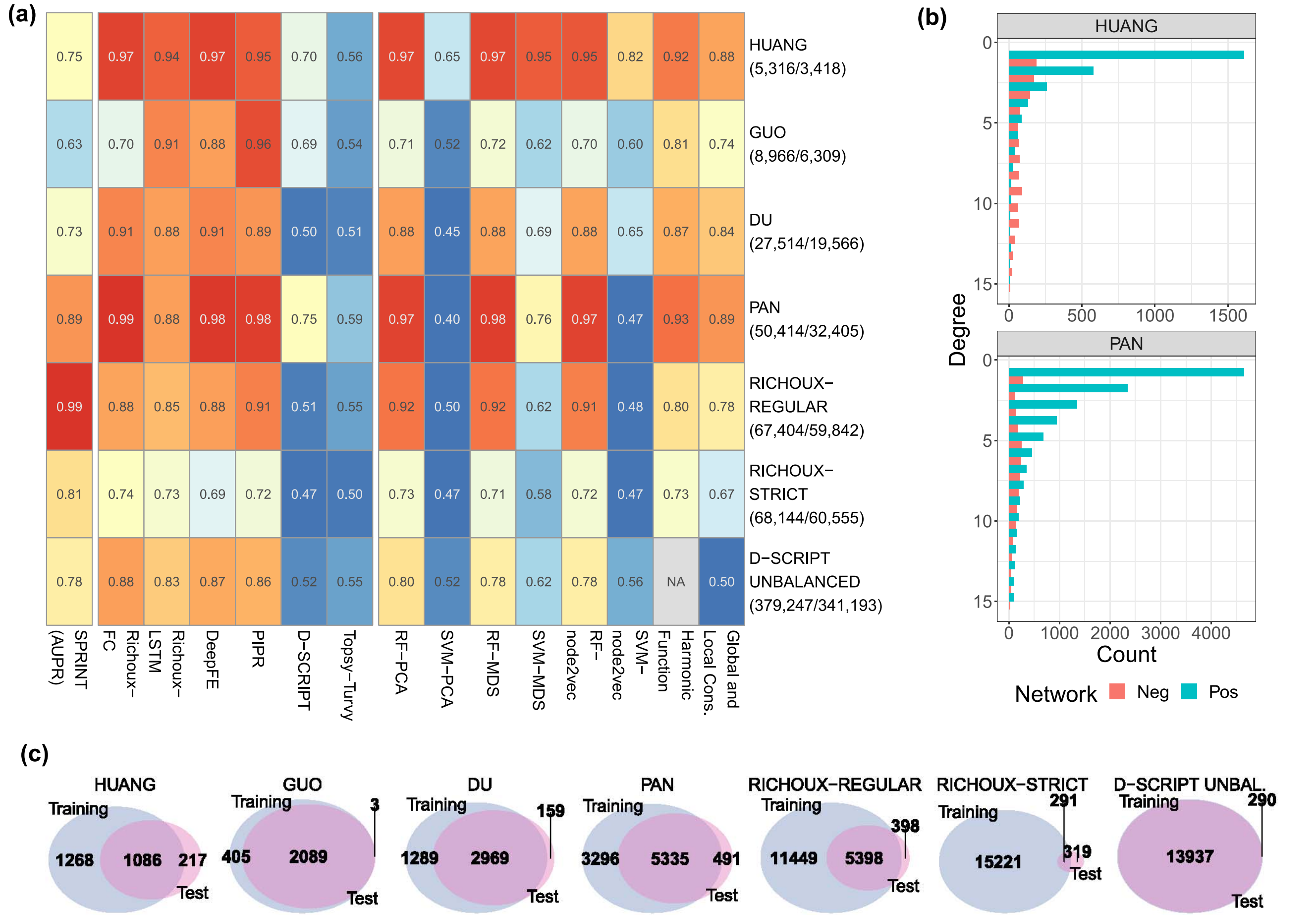

**Figure 2.** Results obtained on randomly split original benchmark datasets. (a) AUPRs on test sets for SPRINT, balanced accuracies for all other methods. Y-axis labels correspond to the dataset; the numbers of samples in the training data (all/restricted length) are shown in parentheses. AUPR values of SPRINT rise with the number of proteins in the dataset. Performances are exceptionally high on the HUANG and PAN dataset. For five out of eight baseline ML models, performances are comparable with the performances of the DL models. All performances drop significantly on RICHOUX-STRICT. (b) Node degree distributions of the positive and negative PPI datasets for HUANG and PAN. Only the node degrees of the positive PPIs follow a power law distribution. (c) Overlap of proteins occurring in the original training and test sets (proteins from positive and negative samples, separate visualization in Supplementary Figure S32).

extremely on a specific subset and will not generalize when presented with unknown proteins.

## Rewiring tests

We investigated how node degree-preserving rewiring of the positive training set affects the methods in general (Explanation 2). The differences to the results on the original datasets are shown in Figure 3(a). As edges are no longer biologically meaningful, the methods can only utilize degree information shortcuts to make correct predictions in the unperturbed test sets. Suppose, in the training split, protein $p$ is only involved in positive (or negative) interactions. Any PPI involving $p$ in the test split will likely be predicted as positive (or negative). Sequence similarities can only help when $p'$ is, e.g. similar to a hub protein. Then, $p'$ is more likely to be a hub protein (e.g. because of a shared promiscuous domain). However, sequence similarities hinting at binding sites or interacting domains cannot be used for prediction because of the rewired training data. While a slight drop in accuracy compared with the original performance would support the validity of Explanation 2, a significant drop would indicate that the models do not learn from node degrees alone. The extent of data leakage and the distribution of node degree ratios remain comparable with the original datasets (Figures 3(b), S6b and S7b).

Indeed, the performances fell slightly compared with the results on the original datasets for all methods. Very high accuracies can still be reached on the datasets HUANG and PAN (Supplemental Figure S14). This is in accordance with our observations from Figure 2(b) and our findings on the original datasets. For these two datasets, the node degree distribution of the positive PPIs (power law) is not equal to that of the negative PPIs (uniform). Additionally, the proportion of training proteins with a high or low degree ratio in the positive and negative part of the test fold is again very high (81% and 91% for HUANG and 90% and 88% for PAN, Supplemental Figure S7b). Therefore, the models can fully leverage the degree information shortcut to predict the unrewired test sets. This explanation also concurs with the observation that the sequence similarity-based baseline ML models lose more performance than the topology-based baseline methods on these datasets. In contrast to the topology-based baseline methods, sequence similarity-based baseline methods must implicitly infer the degree information shortcut.

Remarkably, D-SCRIPT gains some performance on the HUANG dataset and the loss and accuracy curves indicate some learning. Richoux-LSTM has the largest gain in performance on the PAN

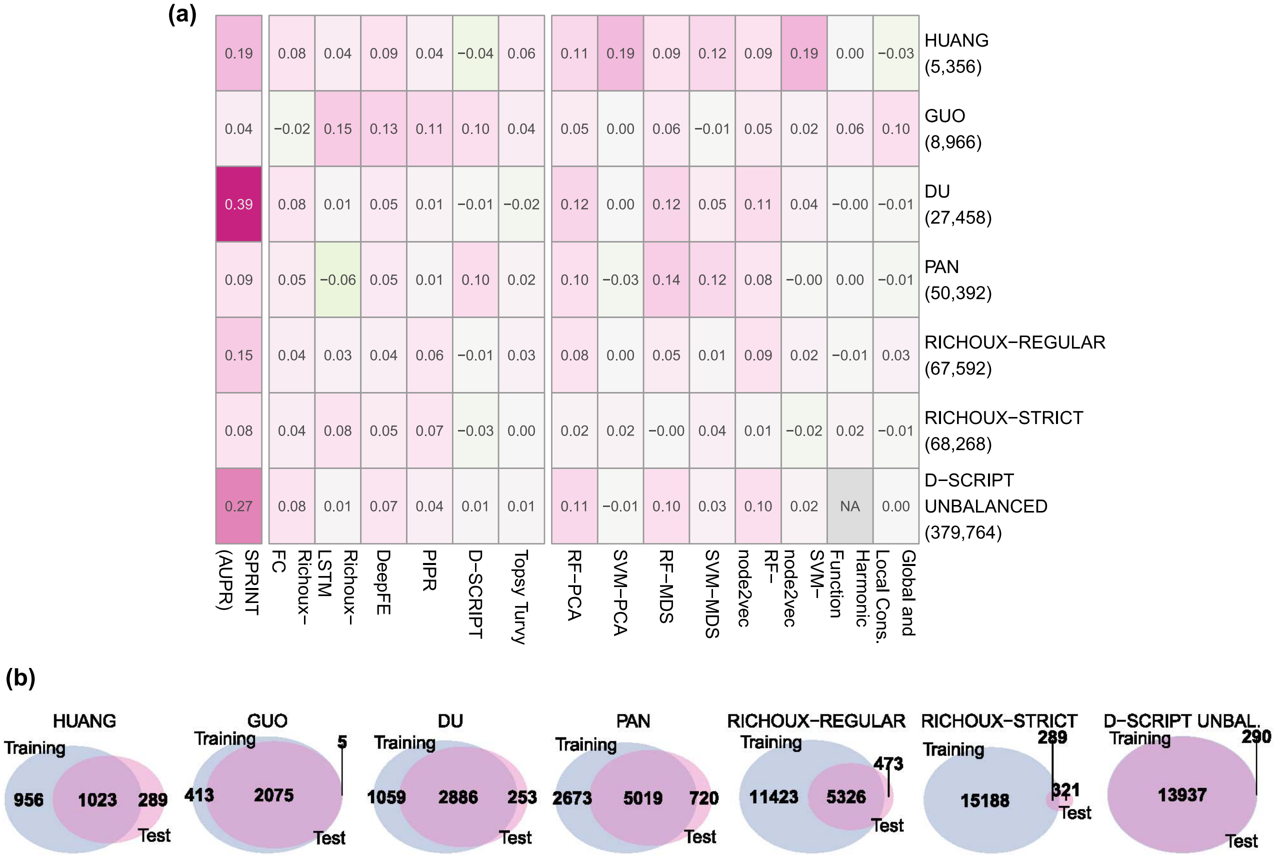

**Figure 3.** Differences between the results obtained on the original datasets and results obtained on datasets with randomly rewired positive PPIs in the train sets. (a) AUPR values on unmodified test sets for SPRINT, balanced accuracies for all other methods. Y-axis labels correspond to the dataset; the numbers of samples in the training data (all/restricted length) are shown in parentheses. Performances fell only slightly compared with the results on the original datasets. Very high accuracies can still be reached for HUANG and PAN. For larger datasets, basic ML models perform approximately as well as the DL models. SPRINT performs almost randomly on small datasets but still reaches accuracies around 80% on the PAN and RICHOUX datasets. (b) Overlap of proteins occurring in the rewired training and test sets (proteins from positive and negative samples, separate visualization in Supplementary Figure S32).

dataset but here, no learning can be seen and in the early stopping setting, the model from epoch 1 was taken. Generally, the random forest-based baseline models lose more accuracy points than the DL models, except on RICHOUX-REGULAR and RICHOUX-STRICT, where the performance is similar. It is possible that the DL methods are better at recognizing node degree biases compared with basic models, which need larger datasets to achieve this. This trend is best visible in the unbalanced D-SCRIPT dataset (Supplemental Figure S14).

SPRINT shows comparably poor performance on the smaller datasets but achieves an AUPR of up to 84% on large datasets. We cannot fully explain these high AUPR values. SPRINT searches for sequence similarities in potentially interacting protein pairs and does thus not benefit from node degree information. While we see a significant drop in the magnitude of the scores compared with the original datasets (Supplemental Figure S5), scores of interacting proteins are still higher than those of non-interacting proteins.

Overall, we can confirm that the methods are biased by node degree information: Balanced accuracies up to 97% can still be reached despite the rewiring of the training data.

## Partitioning tests

Running the baseline methods on the original datasets was a positive test for Explanations 2 and 3, as we expected similar performance for DL and basic ML models. Indeed, our results confirm these expectations (see Figure 2(a)). The partitioning tests served as a negative test for Explanations 2 and 3. If the explanations were valid, we would expect the performances to drop significantly when the models are trained on $INTRA_0$ and tested on $INTRA_1$. We expected this for both the DL and baseline ML models.

The results of the partitioning tests are shown in Figure 4. Notably, all training dataset sizes were approximately halved because of the partitioning strategy. While *INTER* and $INTRA_0$ are approximately equal in size, $INTRA_1$ is considerably smaller (see Methods for details).

Indeed, we observed random or near-to-random performances for all methods trained on $INTRA_0$ and tested on $INTRA_1$. The results show that when the test sets do not suffer from data leakage, the methods do not learn any higher level features during training that they can apply to unseen data. Instead, models overfit on the interaction patterns of the training proteins. Predictions become random when the test set does not contain these proteins (or highly similar proteins). The topology-based baseline methods predict all candidates to interact, except for the unbalanced dataset, where all are predicted not to interact (see recall and specificity, Supplemental Figures S18, S20). D-SCRIPT and Topsy-Turvy profit from early stopping on the large datasets (Supplementary Figure S15). However, for both PAN and D-SCRIPT UNBALANCED, the model from the first epoch had the best performance on the validation set (Supplementary Figures S12, S13).

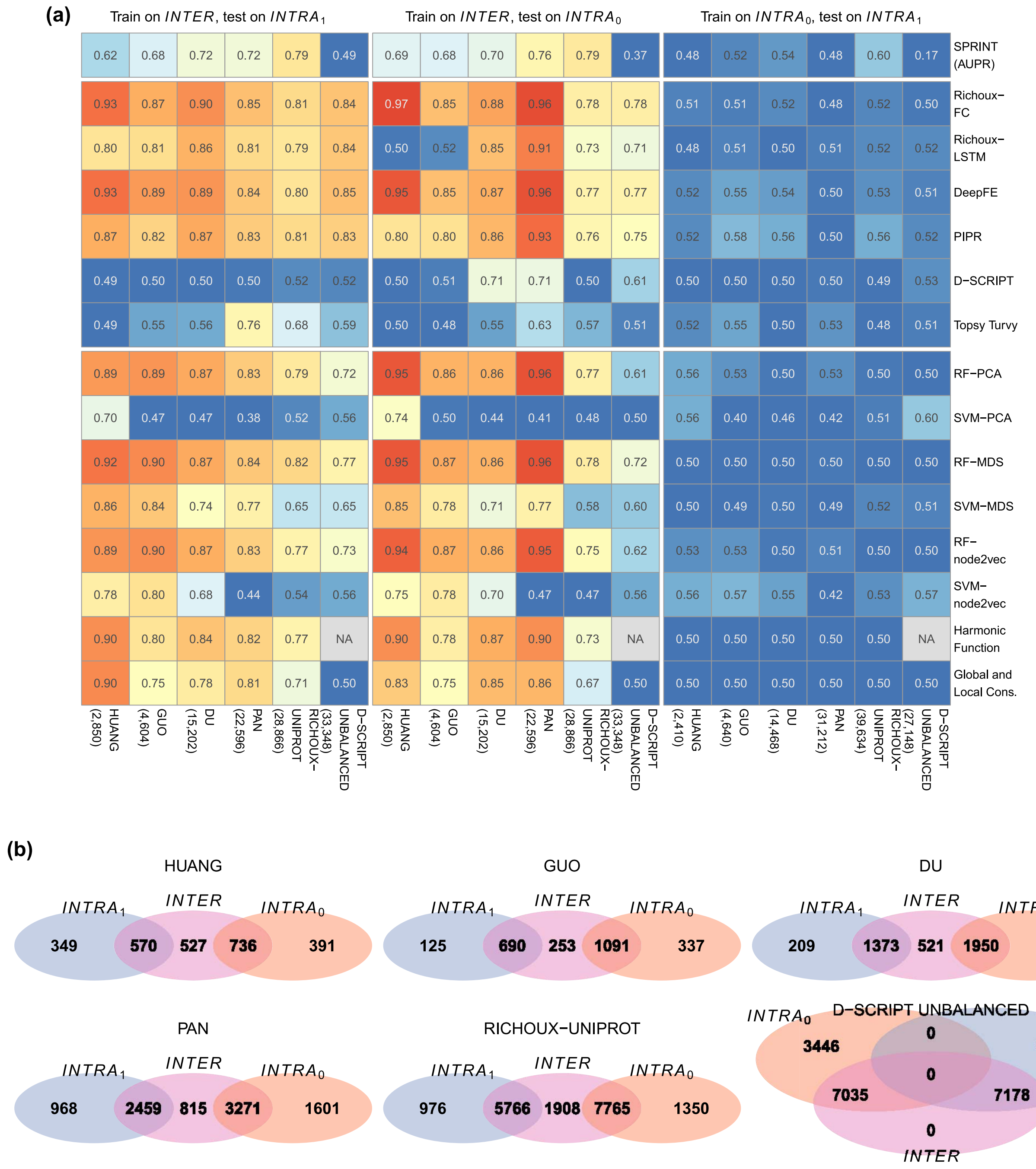

**Figure 4.** Results of partitioning tests. (a) AUPR values for SPRINT, accuracies for all other methods. X-axis labels correspond to the dataset; the numbers of samples in the training data are shown in parentheses. Performances drop to random for all methods when trained on *INTRA*$_0$ and tested on *INTRA*$_1$. Performances obtained from training on *INTER* are still excellent, especially when tested on the *INTRA*$_0$ blocks of HUANG and PAN. (b) Overlap of proteins occurring in the different blocks of the partitions of the benchmark datasets (proteins from positive and negative samples, proteins from positive and negative samples, separate visualization in Supplementary Figure S32).

Overall, the performances obtained after training on *INTER* were excellent, especially when the methods were tested on the *INTRA*$_0$ blocks of HUANG and PAN. Looking at the degree ratio proportions (Supplementary Figure S7), the proportion of *INTER* proteins with a low degree ratio in the negative part of the *INTRA*$_0$ block is remarkably high compared with the *INTRA*$_1$ block. Consequently, the models can leverage node degree information exceptionally well for the non-interactions, reflected in the high specificity achieved in this setting (Supplementary Figure S12.) Richoux-LSTM yielded random predictions for the small datasets due to overfitting, which was foreseeable when running 100 epochs on less than 4000 data points. D-SCRIPT and Topsy-Turvy also show overfitting patterns despite their objectively good performance on PAN and RICHOUX-UNIPROT (Topsy-Turvy, training on *INTER*, testing on *INTRA*$_1$), and DU AND PAN (D-SCRIPT, training on *INTER*, testing on *INTRA*$_0$, Supplementary Figures S12, S13).

Data leakage is highest for the D-SCRIPT UNBALANCED dataset, where all proteins of *INTRA*$_1$ are contained in *INTER*$_1$. This explains why all models perform better on the *INTRA*$_1$ block than

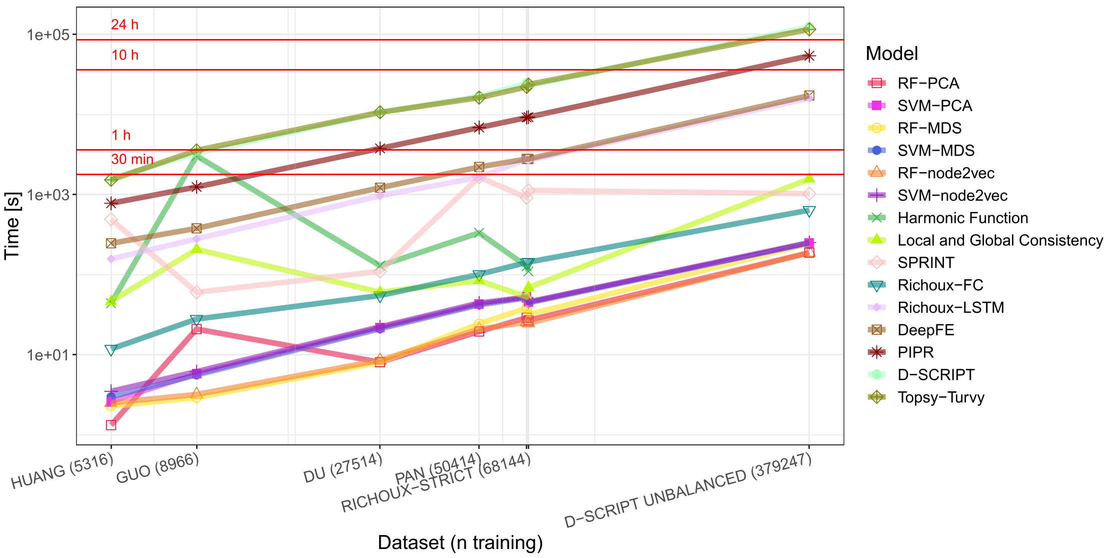

**Figure 5.** Overall runtime for training and testing on the original datasets. The label for RICHOUX-REGULAR is missing from the x-axis because it would overlap with the RICHOUX-STRICT label. Most runtimes increase linearly with size of the training dataset. D-SCRIPT and Topsy-Torvy have by far the highest runtime, while the Random Forest Methods have the lowest.

on the *INTRA*$_0$ block. Looking at the performance developments in the early stopping setting, we can hypothesize that the models find the degree shortcut in the process of overfitting to the training set. Overall, the results obtained show that the effect of the resulting data leakage is considerable even when only one of the proteins of each training PPI occurs in the test set.

## Runtime

Runtime typically grows linearly with the size of the training dataset (Figure 5). An exception is SPRINT which always loads the preprocessed proteome before reading in the positive training PPIs. As these data contain the computed similar sub-sequences, the file is much more extensive for the human proteome compared to yeast; therefore, SPRINT takes longer on human compared with yeast datasets. Also, the topology-based baseline methods have almost constant runtime. However, the harmonic function required more than 350 Gb of memory for the D-SCRIPT UNBALANCED dataset. D-SCRIPT and Topsy-Turvy have by far the highest runtime. Our baseline ML models are the fastest methods. The fastest DL method is Richoux-FC, which only runs for 25 epochs instead of 100, like Richoux-LSTM.

Combined with the results from the previous sections, it is evident that an enormous amount of resources can be saved using simple methods to predict PPIs like scoring algorithms, Random Forests or multilayer perceptrons with few layers. These methods achieve similar results as the costly deep neural networks and can confidently predict PPIs using sequence similarities and learned interaction patterns for known proteins.

All DL models were trained using a single NVIDIA A40 GPU (48 GB). All baseline models were run using standard parameters on a single CPU (Intel® Xeon® Gold 6148) except for the Random Forest baselines (*n_jobs* = 6).

## Gold standard dataset

We showed that existing DL models fail to extract more complex sequence features for predicting PPIs. For the design of more advanced ML strategies, we provide a leakage-free human gold standard data set for training, validation and testing (available at https://doi.org/10.6084/m9.figshare.21591618.v3) [48]. The positive dataset was created using data from HIPPIEv2.3 [49]. Negative PPIs were sampled randomly, but such that individual node degrees are preserved in expectation (Supplemental Figure S3). This dataset was split using our partitioning strategy with KaHIP, i.e. there are no overlaps between the three sets and sequence similarity is minimal. Both training and validation datasets are large enough to allow DL methods to avoid overfitting. Additionally, the sets are redundancy-reduced w.r.t. pairwise sequence similarity using CD-HIT at a 40% threshold [50]. As a result, proteins are also pairwise dissimilar within their set, such that models have to extract features beyond sequence similarity to achieve good performance.

To confirm that our gold standard data set shows the expected behavior, we evaluated all methods on it. Because SPRINT and our baseline models do not have any tunable parameters, we collapsed the training and validation set for their training. The same was done for D-SCRIPT and Topsy-Turvy because they only update their weights using the training dataset. All methods were evaluated on the test set. Indeed, performances were random for all methods (Supplemental Table S5). None of the methods could extract any higher-level features during training that could be applied to predict the test set. Topsy-Turvy was the best-performing method at an accuracy of 56%.

In the early stopping setting, we did not collapse the training and validation set but used the validation dataset to determine the best model. However, no significant changes in performance could be seen. Here, the best-performing method was D-SCRIPT with an accuracy of 55%.

# DISCUSSION

We have conclusively shown that the problem of binary PPI prediction is not solved but wide open. Numerous publications report accuracy values between 90% and 100% and fuel a feedback loop of over-optimism. We have shown that their prediction estimates can be solely attributed to data leakage caused by random splitting into train and test sets. The datasets used in the literature cause the models to overfit based on protein homology and node degree information. More complex sequence features representing binding pockets, protein domains or similar motifs are not extracted. Instead, methods depend on global sequence similarity and node degree.

We have reached our conclusion using three experimental settings: Firstly, we have shown that after random 80/20 splitting of the datasets, DL and baseline ML methods yield interchangeably high results on all datasets. SPRINT, the most straightforward method, performs excellent on large datasets, which shows that finding similar subsequences to predict PPIs is already sufficient to reach exceptional performance measures. When the methods cannot use information about hub proteins (RICHOUX-STRICT test set), performance drops for all methods.

Secondly, we have demonstrated that biologically meaningless edges which preserve the expected node degree do not lead to random predictions. While, for all methods, performance measures fall compared with the original datasets, accuracies up to 97% can still be reached for the DL methods and 89% for the baseline methods. Hence, the models can confidently predict PPIs from node degree information shortcuts only.

Finally, we proved that excluding training proteins from the test set and minimizing pairwise sequence similarities between training and test sets strips all methods of their predictive power. Taken together, our results show that DL methods do not learn any higher level structural features. Conversely, we observe strongly elevated performance scores after training the methods on a data set with shared proteins.

This study has several limitations. Firstly, we focused exclusively on sequence-based methods. In future work, it would be interesting to see if our findings translate to methods predicting PPIs from 3D structures. Since proteins interact in a folded state, methods using this information might extract the actual underlying patterns and find matching sites and domains. There are also algorithmic methods[51, 52] and ML models[5, 18, 41, 53] explicitly relying on phylogeny and co-evolution, whose additional value could be interesting to explore.

Secondly, PPI networks neglect differences in the interactions of protein isoforms [54], where, for instance, the absence of a binding domain will limit a protein's set of interaction partners. Moreover, most proteins work in larger complexes, i.e. they form long-lasting interactions between two or more proteins. These complexes might then perform their functions by interacting transiently with other proteins or protein complexes. If we want to predict and understand the underlying mechanisms of PPIs, we have to consider non-binary interactions and the difference between transient interactions and protein complex formation.

Regarding future directions, we can say that binary PPI predictions are already very accurate for proteins seen during training or for proteins that share similar (sub-)sequences. Hence, we appeal to the community to first try simple methods that cost fewer resources before moving on to complex and deep model architectures. Not only are these models prone to overfitting, but they also waste an unnecessary amount of time, memory, energy and CO2. Training data for species other than yeast or human are currently scarce. Thus, we also see potential in transfer learning approaches as in D-SCRIPT and Topsy-Turvy.

However, the methods we have tested here are not equipped for predicting interactions in the 'dark protein–protein interactome' [55], i.e. currently understudied proteins for which no similar sequences are found in existing PPI networks. Our baseline models fail as they cannot use the similarity shortcut or exploit the network topology resulting from the study bias. We hypothesize that the tested DL models are too simple to learn these complex mechanisms and require significantly more data. The dataset must also be designed to push the models toward learning biological principles instead of shortcuts. We expect that methods leveraging structural information will help to close this gap in the future. As impressively shown by AlphaFold2, DL models have tremendous potential. Similarly to AlphaFold1, D-SCRIPT predicts a contact-map as an intermediate step to predicting interactions. Nevertheless, we observe poor performance for D-SCRIPT and the related Topsy-Turvy, which is only partially improved by early stopping.

We speculate that extensive training data leakage has concealed the full scope of the binary PPI prediction challenge. For future ML efforts, we thus provide a large gold standard training, validation, and test set that is free from data leakage and has minimized pairwise sequence similarities. With this, we hope to kindle renewed interest in this ML challenge and motivate further progress in the refinement of existing PPI prediction networks.

# METHODS

## Datasets

We tested on the seven datasets summarized in Table 1. The yeast dataset GUO [37] contains 5594 positive PPIs from DIP with less than 40% pairwise sequence identity and 5594 negative PPIs generated from pairs of proteins appearing in the positive set, which, according to Swiss-Prot annotations, are expressed in different subcellular locations. The yeast dataset DU [12] was generated similarly and contains 17 257 positive and 48 594 negative PPIs. The human dataset HUANG [40] contains 3899 positive experimentally verified PPIs from HPRD with less than 25% pairwise sequence identity and 4262 negative PPIs, which were generated like the ones of the datasets GUO and DU. The human dataset PAN [38] contains 36 630 positive PPIs from HPRD and 36 480 negative PPIs generated by combining protein pairs obtained via the approach described above with non-interacting pairs contained in the Negatome [56]. The human dataset RICHOUX-REGULAR contains positive PPIs retrieved from UniProt and negative PPIs generated by randomly pairing proteins from the positive set. Sequences were filtered to be at most 1166 amino acids long, mirror copies were added (for each PPI $(p_1, p_2)$, add $(p_2, p_1)$), and the resulting dataset was split into a training ($n_{\text{train}} = 85\,104$), a validation ($n_{\text{val}} = 12\,822$), and a test fold ($n_{\text{test}} = 12\,822$). The human dataset RICHOUX-STRICT [2] was constructed from the RICHOUX-UNIPROT dataset as follows: PPIs whose involved proteins appear less than 3 times were assigned to the test fold. The remainder was redistributed among the training and validation datasets. The resulting sizes of the training, validation, and test folds are, respectively, $n_{\text{train}} = 91\,036$, $n_{\text{val}} = 12\,506$, and $n_{\text{test}} = 720$. The D-SCRIPT UNBALANCED [20] dataset contains 43 128 positive and 431 379 negative PPIs, split into training (38 344 positives / 383 448 negatives) and test set (4794 positives / 47 931 negatives). The positive PPIs are experimentally verified interactions downloaded from STRING, with lengths between 50 and 800 amino acids. Highly redundant sequences ($\geq$ 40% pairwise sequence identity) were removed. Negative PPIs were generated from the positive set at a one to ten ratio to reflect that there are much more non-interacting proteins than interacting proteins.

The seven datasets were cleaned from duplicates and checked for overlaps. The training and validation folds in RICHOUX-REGULAR and RICHOUX-STRICT were joined for all analyses. All datasets except RICHOUX-REGULAR, RICHOUX-STRICT and D-SCRIPT UNBALANCED were randomly split into a train (80%) and

**Table 2:** State of the benchmark datasets after cleaning and balancing: $n$ denotes the overall number of samples in the datasets, i.e. the number of PPIs plus the number of randomly sampled non-edges. $n_{\text{train}}$ and $n_{\text{test}}$ are defined analogously for the train and test sets. The modifications were done to clean and balance the original benchmark datasets, i.e. to ensure that the number of positive PPIs (edges) equals the number of negative PPIs (non-edges) in the train and test splits of all datasets.

| Dataset | $n$ | $n_{\text{train}}$ | $n_{\text{test}}$ | Modifications |
|---|---|---|---|---|
| GUO | 11 162 | 8966 | 2196 | 24duplicates, sampled 35 negatives for training, dropped 37 negatives for testing |
| DU | 34 512 | 27 514 | 6998 | 2511 duplicates, dropped 23 157 negatives for training and 5680 for testing |
| HUANG | 6690 | 5316 | 1374 | 0 duplicates, dropped 619 negatives for training and 110 for testing |
| PAN | 62 962 | 50 414 | 12 548 | 55 duplicates, dropped 1678 negatives for training and 476 for testing |
| RICHOUX-REGULAR | 79 868 | 67 404 | 12 464 | 5047 duplicates, dropped 25 475 negatives for training and 342 for testing |
| RICHOUX-STRICT | 68 664 | 68 144 | 520 | 5341 duplicates, dropped 30 057 negatives for training and 200 for testing |
| D-SCRIPT UNBAL. | 426 492 | 379 247 | 47 245 | 57 duplicates, sampled 14 081 negatives for training, sampled 1660 negatives for testing |

**Table 3:** State of the benchmark datasets after rewiring the positive training PPIs and balancing the datasets. $n$ denotes the overall number of samples in the datasets, i.e. the number of PPIs plus the number of randomly sampled non-edges. $n_{\text{train}}$ and $n_{\text{test}}$ are defined analogously for the train and test sets.

| Dataset | $n$ | $n_{\text{train}}$ | $n_{\text{test}}$ | Modifications |
|---|---|---|---|---|
| GUO | 11 256 | 8966 | 2290 | 24duplicates, dropped 12 negatives for training, sampled 57 negatives for testing |
| DU | 34 416 | 27 458 | 6958 | 2511 duplicates, dropped 23086 negatives for training and 5711 for testing |
| HUANG | 6722 | 5356 | 1366 | 0 duplicates, dropped 595 negatives for training and 118 for testing |
| PAN | 62 974 | 50 392 | 12 582 | 55 duplicates, dropped 1706 negatives for training and 442 for testing |
| RICHOUX-REGULAR | 80 056 | 67 592 | 12464 | 5047 duplicates, dropped 25 382 negatives for training and 342 for testing |
| RICHOUX-STRICT | 68 788 | 68 268 | 520 | 5341 duplicates, dropped 29 996 negatives for training and 200 for testing |
| D-SCRIPT UNBAL. | 427 009 | 379 764 | 47245 | 57 duplicates, sampled 14 832 negatives for training and 1660 for testing |

test (20%) set. A validation set was not needed since we omitted hyperparameter optimization. For the early stopping setting, we used 10% of the train set and a patience of 5. This set was used to determine the model with the best validation accuracy (precision for DeepFE since the method was optimized for that in the original publication), which was later used to predict the test set. After splitting, the datasets were balanced either by randomly dropping negatives or by sampling new negatives such that both proteins are already part of the dataset and that the interaction is not part of the existing positive or negative interactions (Table 2). The imbalance of the D-SCRIPT UNBALANCED dataset was maintained to test its influence on method performance.

Because of GPU restrictions, we created length-restricted versions for all datasets for D-SCRIPT and Topsy-Turvy, where each protein's length was restricted to lie between 50 and 1000 amino acids (Table 1).

## Rewiring tests

In order to test how much the models learn from node degree only, we rewired the positive PPIs of all described training datasets such that all proteins keep their original degree in expectation (see Figure 6). However, the edges are newly assigned, rendering them biologically meaningless. For this, we used the `expected_degree_graph()` function of the NetworkX 2.8 Python package. Given the list $(w_0, w_1, \ldots, w_{m-1})$ of node degrees in the original network (positive PPIs in training fold), the function constructs a graph with $m$ nodes and assigns an edge between node $u$ and node $v$ with probability $p_{uv} = \frac{w_u w_v}{\sum_k w_k}$. Again, all datasets were checked for duplicates and overlaps and were balanced after splitting, resulting in the counts summarized in Table 3.

A significant drop in accuracy compared with the performance on the original dataset could indicate that the models learn from the sequence features. However, a small drop would indicate that the models mostly memorize node degrees and assign their predictions based on whether or not the protein is overall likely to interact (Explanation 2).

## Partitioning tests

To explore Explanations 2 and 3, which hypothesize that the models mostly learn from node degree information shortcuts and sequence similarities (see Introduction), we partitioned the yeast and human proteomes into two disjoint subsets $P_0$ and $P_1$ such that proteins from different subsets are pairwise unsimilar. For this, we first exported the yeast and human similarity networks by SIMAP2 as METIS files with length-normalized bitscore weights:

$$w_{p1,p2} = n^{-1} \cdot \sum_{i=1}^{n} \mathit{length}(p_i) \cdot \frac{\mathit{bitscore}(p_1, p_2)}{\min\{\mathit{length}(p_1), \mathit{length}(p_2)\}} \tag{1}$$

This resulted in weighted similarity networks with, respectively, 6718 nodes and 92 409 edges (for the yeast proteome) and 20 353 nodes and 1 900 490 edges (for the human proteome). In the similarity networks, bitscore edge weights increase with increasing pairwise sequence similarity.

These similarity networks were then given to the KaHIP KaFFPa algorithm (desired output partitions: 2, pre-configuration: strong), which (heuristically) solves the following problem: Given a graph $G = (V, E, \omega)$ with non-negative edge weights $\omega : E \to \mathbb{R}_{\geq 0}$, it partitions $V$ into blocks $P_0$ and $P_1$ such that, for all $i \in \{0, 1\}$, it holds that $|P_i| \leq (1+\epsilon)\lceil \frac{|V|}{2} \rceil$ (partition is almost balanced) and the total cut size $\omega(P_0, P_1) = \sum_{u \in P_0} \sum_{v \in P_1} \omega(uv) \cdot [uv \in E]$ is minimized (the hyperparameter $\epsilon$ was left at the default $\epsilon = 0.03$). For both the yeast and the human proteome, we hence obtained two disjoint subsets of proteins such that the overall pairwise sequence similarity between the subsets (sum of normalized bitscores along the cut) is minimized.

For the yeast proteome, this resulted in $|P_0| = 3458$ and $|P_1| = 3260$; for the human proteome, we obtained $|P_0| = 10481$ and

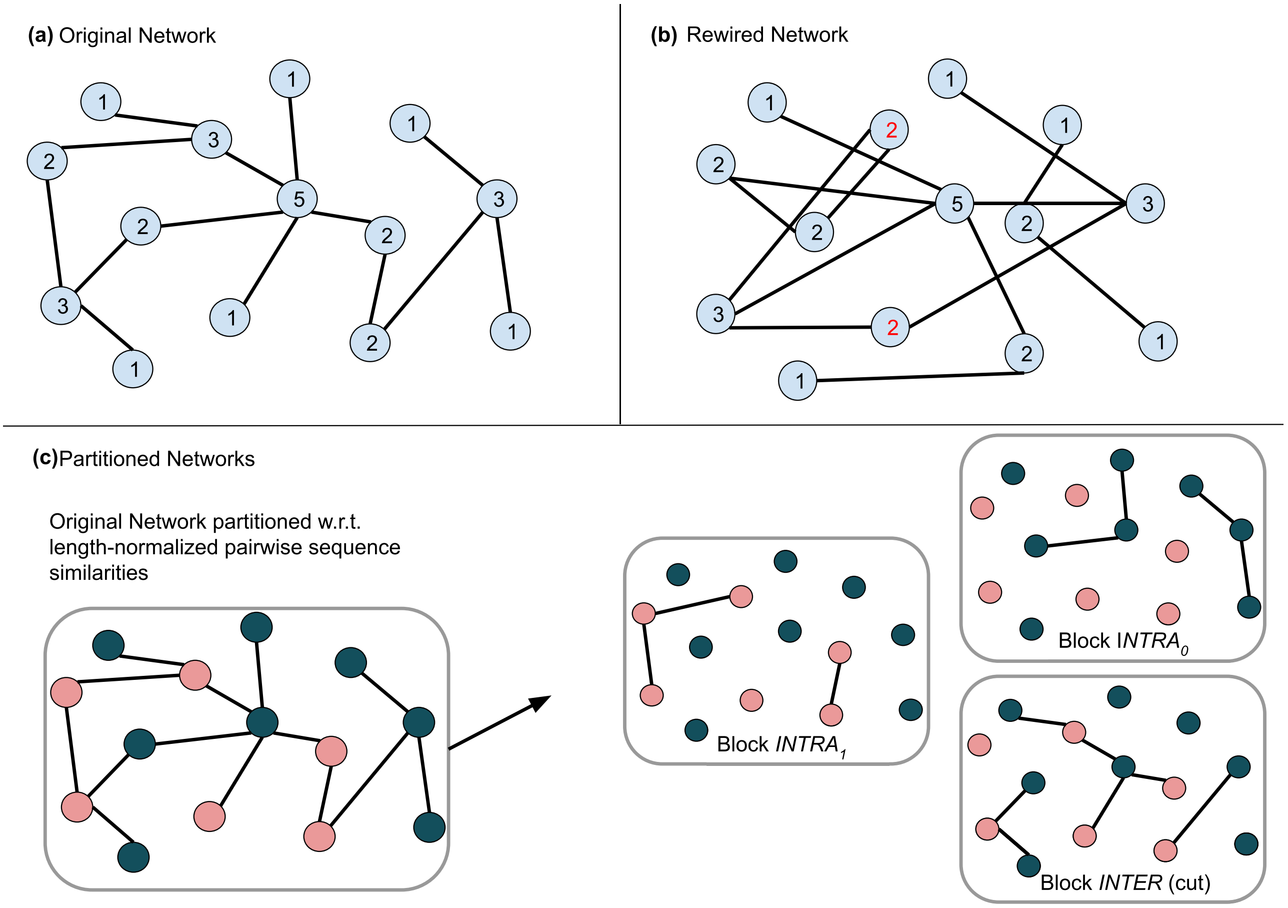

**Figure 6.** Concept of the rewiring and partitioning strategies. When the original network (a) is rewired, all node degrees stay the same in expectation but the edges are no longer meaningful (b). For the partitioning tests (c), the proteome is partitioned into two blocks (pink and green nodes) such that pairwise inter-block sequence similarity is minimized. Then, the PPIs from the original network (a) are partitioned based on which block the involved proteins are contained in (*INTRA*$_0$: both proteins contained in green block, *INTRA*$_1$: both proteins in contained pink block, *INTER*: proteins contained in different blocks).

$|P_1| = 9872$. Based on the partition $\{P_0, P_1\}$ of the human and yeast proteomes, we then partitioned each PPI dataset into three blocks *INTRA*$_0$, *INTRA*$_1$, and *INTER*. Each PPI $(p_1, p_2)$ was assigned to one of these blocks as follows:

- We assigned $(p_1, p_2)$ to the block *INTRA*$_0$ if $p_1, p_2 \in P_0$.
- We assigned $(p_1, p_2)$ to the block *INTRA*$_1$ if $p_1, p_2 \in P_1$.
- We assigned $(p_1, p_2)$ to the block *INTER* if $p_1 \in P_0 \wedge p_2 \in P_1$ or $p_1 \in P_1 \wedge p_2 \in P_0$.

Again, all datasets were cleaned from duplicates and balanced after partitioning. If additional negatives had to be sampled, they were sampled from the proteins of the respective block. This yielded the number of samples shown in Table 4. Methods were then either trained on block *INTRA*$_0$ and tested on block *INTRA*$_1$ or trained on block *INTER* and tested on the blocks *INTRA*$_0$ and *INTRA*$_1$. Following Explanations 2 and 3, we expected the most significant drop in accuracy compared with the original performance when training on *INTRA*$_0$ and testing *INTRA*$_1$. We expected a smaller drop in performance when training block *INTER* and testing on *INTRA*$_0$ and *INTRA*$_1$, since then, for approximately half of the test PPIs, sequence similarity information and node degrees from training are available at test time. Note that, from the three datasets published by Richoux *et al.* [2], we partitioned the dataset RICHOUX-UNIPROT as it contains the largest number of unique proteins.

## Construction of gold standard dataset

The whole human proteome was split into three parts by running KaHIP on the all-against-all sequence similarity matrix from SIMAP2 with length-normalized bitscores. When configured to output a three-way partition, KaHIP partitions the node set V of an edge-weighted graph $G = (V, E, \omega)$ into blocks $P_0$, $P_1$ and $P_2$ such that the cut size

$$\omega(P_0, P_1, P_2) = \sum_{\{i,j\} \in \binom{\{0,1,2\}}{2}} \sum_{u \in P_i} \sum_{v \in P_j} \omega(uv) \cdot [uv \in E] \qquad (2)$$

is minimized and $|P_i| \leq (1+\epsilon)\lceil \frac{|V|}{3} \rceil$ holds for all $i \in \{0, 1, 2\}$. This resulted in 6987 proteins in $P_0$, 6987 proteins in in $P_1$ and 6379 proteins in $P_2$.

A total of 831 933 positive PPIs were downloaded from the HIPPIE database [49] (version 2.3). Mapping all 18 909 unique IDs to UniProt IDs using the UniProt mapping tool resulted in 17 269 unique proteins and 689 735 PPIs. The positive dataset was sorted into blocks *INTRA*$_0$ (56 747 PPIs), *INTRA*$_1$ (164 416 PPIs) and *INTRA*$_2$ (52 560 PPIs), where $uv \in$ *INTRA*$_i$ if and only if $u \in P_i$ and $v \in P_i$. Negative PPIs were sampled randomly to match the number of positives. To exclude the possibility of learning from node degrees alone, we approximately preserved the node degrees of the proteins from the positive networks *INTRA*$_i$ in the negative

**Table 4:** Number of samples contained in each block after splitting the benchmark datasets according to the partitioning assignments. $n_0$, $n_1$ and $n_{INTER}$ denote the numbers of positive and negative PPIs in the blocks *INTRA*$_0$, *INTRA*$_1$ and *INTER*. All blocks are balanced (50% interactions, 50% non-interactions).

| Dataset | $n_0$ | $n_1$ | $n_{INTER}$ | Modifications |
|---|---|---|---|---|
| GUO | 4640 | 1722 | 4604 | *INTRA*$_0$: 218 additional negatives; *INTRA*$_1$: 86 deleted negatives; *INTER*: 307 deleted negatives |
| DU | 14 468 | 4842 | 15 202 | *INTRA*$_0$: 9686 deleted negatives; *INTRA*$_1$: 4707 deleted negatives; *INTER*: 14 357 deleted negatives |
| HUANG | 2410 | 1426 | 2850 | *INTRA*$_0$: 245 additional negatives; *INTRA*$_1$: 66 deleted negatives; *INTER*: 544 deleted negatives |
| PAN | 31 212 | 9150 | 22 596 | *INTRA*$_0$: 180 additional negatives; *INTRA*$_1$: 1422 additional negatives; *INTER*: 3601 deleted negatives |
| RICHOUX-UNIPROT | 39 634 | 10 334 | 28 866 | *INTRA*$_0$: 9323 additional negatives; *INTRA*$_1$: 4997 deleted negatives; *INTER*: 6124 deleted negatives |
| D-SCRIPT UNBAL. | 149 314 | 93 137 | 183 414 | *INTRA*$_0$: 50 730 additional negatives; *INTRA*$_1$: 15929 deleted negatives; *INTER*: 18 772 deleted negatives |

networks. This was achieved by randomly sampling two distinct proteins at a time from the multiset

$$M_i = \{\, \underbrace{p_1, \dots p_1}_{\boldsymbol{deg}_i(p_1)\text{ times}}, \dots, \underbrace{p_k, \dots p_k}_{\boldsymbol{deg}_i(p_k)\text{ times}}, \dots \mid p_k \in P_i \}, \qquad (3)$$

where the number of occurences of each protein $p_k \in P_i$ equals its degree $\boldsymbol{deg}_i(p_k)$ in *INTRA*$_i$.

Afterward, the sequences of the individual proteins in the blocks were fed to CD-HIT at a similarity threshold of 40%. Within *INTRA*$_0$, CD-HIT identified 1512 redundant sequences, within *INTRA*$_1$ 1680, and within *INTRA*$_2$ 1465. Between *INTRA*$_0$ and *INTRA*$_1$, CD-HIT 2D found three redundant sequences, between *INTRA*$_0$ and *INTRA*$_2$ 20 and between *INTRA*$_1$ and *INTRA*$_2$ 24. These sequences were filtered out of the blocks to form redundancy-reduced datasets. The blocks were then balanced again, resulting in 59 260 PPIs in *INTRA*$_0$, 163 192 PPIs in *INTRA*$_1$ and 52 048 PPIs in *INTRA*$_2$. Finally, we labeled the block *INTRA*$_1$ as the training dataset, the block *INTRA*$_0$ as the validation dataset and the block *INTRA*$_2$ as the test dataset.

## Tested methods

The results of an extensive literature screening for high-performing PPI prediction methods can be found in Supplemental Table S1. Only 12 of 32 reviewed publications made their code available (13 methods; Richoux *et al.* [2] proposed two). We excluded the methods that did not only use sequences as input and focused on DL methods with high reported accuracies, which we managed to reproduce with reasonable effort. Additionally, we included the SPRINT method as a baseline comparison since it only relies on sequence similarity for its predictions. Further details about the tested methods can be found in the Supplementary Material.

We further included simple baseline ML methods, which we designed such that they can only learn from sequence similarity. For this, we encoded amino acid sequences as vectors of sequence similarities to all other proteins in the human or yeast proteome. We reduced the dimensionality of the similarity-based encodings using PCA, MDS and node2vec, and then trained random forests and SVMs on the dimensionality-reduced encodings. We also included two classical node label classification algorithms (harmonic function [34], global and local consistency [57]), which we ran on the line graphs of the PPI networks. Conversely to the similarity-based baselines, these methods do not use sequence information at all but predict interactions only based on the topology of the network induced by the positive and negative PPIs in the training data. Note that we did not include these baselines in order to test if simple methods are sufficient for PPI prediction but in order to quantify possible data leakage to due sequence similarity or network topology (good performance of the baselines is indicative of data leakage). In order to minimize the risk of confirmation bias, we therefore consciously decided not to carry out hyper-parameter optimization and to use classical methods that are implemented in very popular software packages (scikit-learn for random forests and SVMs and NetworkX for harmonic function and global and local consistency).

In the main figures, we report balanced accuracy for all methods except for SPRINT and include other performance measures in the Supplement. SPRINT calculates similarity scores and sorts the output decreasingly by the scores where a higher score represents a higher probability for interaction. Rather than choosing an arbitrary threshold to calculate accuracies, we calculated the AUC and auPR for SPRINT.

# AUTHOR CONTRIBUTIONS STATEMENT

J.B., D.B.B. and M.L. designed and conceived this study and drafted the manuscript. J.B. implemented the test protocol and carried out the analyses. M.L. and D.B.B. supervised this work.

# ACKNOWLEDGMENTS

We thank Prof. Dr. Thomas Rattei and his team from SIMAP2 at the University of Vienna for kindly and quickly sharing their protein similarity data. J.B. and M.L. were supported by the German Federal Ministry of Education and Research (BMBF) within the framework of the CompLS funding concept [031L0305A (DROP2AI)]. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. M.L. was additionally funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) [422216132]. D.B.B. was supported by the German Federal Ministry of Education and Research (BMBF) within the framework of the CompLS funding concept [031L0309A (NetMap)].

# CODE AND DATA AVAILABILITY

All datasets and code are available at https://github.com/biomedbigdata/data-leakage-ppi-prediction. The gold standard dataset is available at https://doi.org/10.6084/m9.figshare.21591618.v3. An AIMe report [58] specifying the details of all analyses is available at https://aime-registry.org/report/VRPXym.

# REFERENCES

1. Srinivasa Rao V, Srinivas K, Sujini GN, Kumar GN. Protein-protein interaction detection: methods and analysis. *Int J Proteomics* 2014;**2014**:1–12.

2. Richoux F, Servantie C, Borès C, Téletchéa S. Comparing two deep learning sequence-based models for protein-protein interaction prediction. *arXiv preprint arXiv:190106268* 2019.
3. Sun T, Zhou B, Lai L, Pei J. Sequence-based prediction of protein protein interaction using a deep-learning algorithm. *BMC Bioinformatics* 2017;**18**(1):1–8.
4. Chen M, Ju CJ-T, Zhou G, et al. Multifaceted protein–protein interaction prediction based on siamese residual rcnn. *Bioinformatics* 2019a;**35**(14):i305–14.
5. Wang L, Wang H-F, Liu S-R, et al. Predicting protein-protein interactions from matrix-based protein sequence using convolution neural network and feature-selective rotation forest. *Sci Rep* 2019;**9**(1):1–12.
6. Da X, Hanxiao X, Zhang Y, et al. Protein-protein interactions prediction based on graph energy and protein sequence information. *Molecules* 1841;**25**(8):2020.
7. Wang J, Zhang L, Jia L, et al. Protein-protein interactions prediction using a novel local conjoint triad descriptor of amino acid sequences. *Int J Mol Sci* 2017;**18**(11):2373.
8. You Z-H, Chan KCC, Pengwei H. Predicting protein-protein interactions from primary protein sequences using a novel multi-scale local feature representation scheme and the random forest. *PloS One* 2015a;**10**(5):e0125811.
9. You Z-H, Li J, Gao X, et al. Detecting protein-protein interactions with a novel matrix-based protein sequence representation and support vector machines. *Biomed Res Int* 2015;**2015**:1–9.
10. You Z-H, Lei Y-K, Zhu L, et al. Prediction of protein-protein interactions from amino acid sequences with ensemble extreme learning machines and principal component analysis. *BMC Bioinformatics* 2013;**14**(8):1–11.
11. Lun H, Chan KCC. Discovering variable-length patterns in protein sequences for protein-protein interaction prediction. *IEEE Trans Nanobioscience* 2015;**14**(4):409–16.
12. Xiuquan D, Sun S, Changlin H, et al. Deepppi: boosting prediction of protein–protein interactions with deep neural networks. *J Chem Inf Model* 2017;**57**(6):1499–510.
13. Yao Y, Xiuquan D, Diao Y, Zhu H. An integration of deep learning with feature embedding for protein–protein interaction prediction. *PeerJ* 2019;**7**:e7126.
14. Jha K, Saha S. Amalgamation of 3d structure and sequence information for protein–protein interaction prediction. *Sci Rep* 2020;**10**(1):1–14.
15. Saha I, Zubek J, Klingström T, et al. Ensemble learning prediction of protein–protein interactions using proteins functional annotations. *Mol Biosyst* 2014;**10**(4):820–30.
16. Chen K-H, Wang T-F, Yuh-Jyh H. Protein-protein interaction prediction using a hybrid feature representation and a stacked generalization scheme. *BMC Bioinformatics* 2019b;**20**(1):1–14.
17. Zhao L, Wang J, Yang H, Cheng L. Conjoint feature representation of go and protein sequence for ppi prediction based on an inception rnn attention network. *Molecular Therapy-Nucleic Acids* 2020;**22**:198–208.
18. Hashemifar S, Neyshabur B, Khan AA, Jinbo X. Predicting protein–protein interactions through sequence-based deep learning. *Bioinformatics* 2018;**34**(17):i802–10.
19. Maetschke SR, Simonsen M, Davis MJ, Ragan MA. Gene ontology-driven inference of protein–protein interactions using inducers. *Bioinformatics* 2012;**28**(1):69–75.
20. Sledzieski S, Singh R, Cowen L, Berger B. D-script translates genome to phenome with sequence-based, structure-aware, genome-scale predictions of protein-protein interactions. *Cell Syst* 2021;**12**(10):969–982.e6.
21. Singh R, Devkota K, Sledzieski S, et al. Topsy-turvy: integrating a global view into sequence-based ppi prediction. *Bioinformatics* 2022;**38**(Supplement_1):i264–72.
22. Khatun M, Watshara Shoombuatong M, Hasan HK, et al. Evolution of sequence-based bioinformatics tools for protein-protein interaction prediction. *Curr Genomics* 2020;**21**(6):454–63.
23. Jumper J, Evans R, Pritzel A, et al. Highly accurate protein structure prediction with alphafold. *Nature* 2021;**596**(7873):583–9.
24. Evans R, O'Neill M, Pritzel A, et al. Protein complex prediction with alphafold-multimer. *BioRxiv* 2021.
25. Park Y, Marcotte EM. Flaws in evaluation schemes for pair-input computational predictions. *Nat Methods* 2012;**9**(12):1134–6.
26. Hamp T, Rost B. More challenges for machine-learning protein interactions. *Bioinformatics* 2015a;**31**(10):1521–5.
27. Li S, Sanan W, Wang L, et al. Recent advances in predicting protein–protein interactions with the aid of artificial intelligence algorithms. *Curr Opin Struct Biol* 2022;**73**:102344.
28. Whalen S, Schreiber J, Noble WS, Pollard KS. Navigating the pitfalls of applying machine learning in genomics. *Nat Rev Genet* 2022;**23**(3):169–81.
29. Kapoor S, Narayanan A. Leakage and the reproducibility crisis in ML-based science. *arXiv preprint arXiv:220707048* 2022.
30. Kaufman S, Rosset S, Perlich C, Stitelman O. Leakage in data mining: formulation, detection, and avoidance. *ACM Trans Knowl Discov Data* 2012;**6**(4):1–21.
31. Ayan Chatterjee, Robin Walters, Zohair Shafi, Omair Shafi Ahmed, Michael Sebek, Deisy Gysi, Rose Yu, Tina Eliassi-Rad, Albert-László Barabási, Giulia Menichetti, Omair Shafi Ahmed, Michael Sebek, Deisy Gysi, Rose Yu, Tina Eliassi-Rad, Albert-László Barabási, and Giulia Menichetti. Improving the generalizability of protein-ligand binding predictions with ai-bind *Nat Commun* 2023;**14**(1):1989.
32. Ben-Hur A, Noble WS. Choosing negative examples for the prediction of protein-protein interactions. *BMC Bioinformatics* 2006;**7**(1):1–6.
33. Berggård T, Linse S, James P. Methods for the detection and analysis of protein–protein interactions. *Proteomics* 2007;**7**(16):2833–42.
34. Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In: Tom Fawcett, Nina Mishra, (eds.) *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, Washington, DC, USA: AAAI Press, 2003.
35. Zhou D, Bousquet O, Lal T, et al. Learning with local and global consistency. *Adv Neural Inf Process Syst* 2003a;**16**.
36. Li Y, Ilie L. Sprint: ultrafast protein-protein interaction prediction of the entire human interactome. *BMC Bioinformatics* 2017;**18**(1):1–11.
37. Guo Y, Lezheng Y, Wen Z, Li M. Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences. *Nucleic Acids Res* 2008;**36**(9):3025–30.
38. Pan X-Y, Zhang Y-N, Shen H-B. Large-scale prediction of human protein- protein interactions from amino acid sequence based on latent topic features. *J Proteome Res* 2010;**9**(10):4992–5001.
39. Arnold R, Goldenberg F, Mewes H-W, Rattei T. Simap-the database of all-against-all protein sequence similarities and annotations with new interfaces and increased coverage. *Nucleic Acids Res* 2014;**42**(D1):D279–84.
40. Huang Y-A, You Z-H, Gao X, et al. Using weighted sparse representation model combined with discrete cosine transformation to predict protein-protein interactions from protein sequence. *Biomed Res Int* 2015;**2015**:1–10.

41. Wang L, You Z-H, Yan X, et al. Using two-dimensional principal component analysis and rotation forest for prediction of protein-protein interactions. *Sci Rep* 2018;**8**(1):1–10.
42. Ding Y, Tang J, Guo F. Predicting protein-protein interactions via multivariate mutual information of protein sequences. *BMC Bioinformatics* 2016;**17**(1):1–13.
43. Guo Y, Li M, Xuemei P, et al. Pred_ppi: a server for predicting protein-protein interactions based on sequence data with probability assignment. *BMC Res Notes* 2010;**3**(1):1–7.
44. Shen J, Zhang J, Luo X, et al. Predicting protein–protein interactions based only on sequences information. *Proc Natl Acad Sci* 2007;**104**(11):4337–41.
45. Satyajit Mahapatra, Anish Kumar, Animesh Sharma, and Sitanshu Sekhar Sahu. Effect of dimensionality reduction on classification accuracy for protein–protein interaction prediction. In: Bibudhendu Pati, Chhabi Rani Panigrahi, Rajkumar Buyya, Kuan-Ching Li, (eds.) *Advanced Computing and Intelligent Engineering*, pages 3–12. Singapore: Springer Singapore, 2020.
46. Ieremie I, Ewing RM, Niranjan M. Transformergo: predicting protein–protein interactions by modelling the attention between sets of gene ontology terms. *Bioinformatics* 2022;**38**(8): 2269–77.
47. Peter Sanders and Christian Schulz. Think locally, act globally: Highly balanced graph partitioning. In: Vincenzo Bonifaci, Camil Demetrescu, Alberto Marchetti-Spaccamela, (eds.) *International Symposium on Experimental Algorithms*, pages 164–75. Germany: Springer Berlin-Heidelberg, 2013.
48. Bernett J. PPI prediction from sequence, gold standard dataset. *figshare* 2022; URL https://figsharecom/articles/dataset/PPI_prediction_from_sequence_gold_standard_dataset/21591618.
49. Alanis-Lobato G, Andrade-Navarro MA, Schaefer MH. Hippie v2. 0: enhancing meaningfulness and reliability of protein–protein interaction networks. *Nucleic Acids Res* 2016;**45**:D408–14.
50. Limin F, Niu B, Zhu Z, et al. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* 2012 . ISSN 1367-4803;**28**(23):3150–2. 10.1093/bioinformatics/bts565.
51. Pazos F, Valencia A. Similarity of phylogenetic trees as indicator of protein–protein interaction. *Protein Eng* 2001;**14**(9): 609–14.
52. Ochoa D, Juan D, Valencia A, Pazos F. Detection of significant protein coevolution. *Bioinformatics* 2015;**31**(13):2166–73.
53. Hamp T, Rost B. Evolutionary profiles improve protein–protein interaction prediction from sequence. *Bioinformatics* 2015b;**31**(12):1945–50.
54. Louadi Z, Yuan K, Gress A, et al. Digger: exploring the functional role of alternative splicing in protein interactions. *Nucleic Acids Res* 2021;**49**(D1):D309–18.
55. Tabar MS, Parsania C, Chen H, et al. Illuminating the dark protein-protein interactome. *Cell reports. Methods* 2022; **2**(8):100275.
56. Blohm P, Frishman G, Smialowski P, et al. Negatome 2.0: a database of non-interacting proteins derived by literature mining, manual annotation and protein structure analysis. *Nucleic Acids Res* 2014;**42**(D1):D396–400.
57. Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8–13, 2003, Vancouver and Whistler, British Columbia, Canada]*, pages 321–328. Cambridge, MA, USA: MIT Press, 2003b. URL https://proceedings.neurips.cc/paper/2003/hash/87682805257e619d49b8e0dfdc14affa-Abstract.html.
58. Matschinske J, Alcaraz N, Benis A, et al. The AIMe registry for artificial intelligence in biomedical research. *Nat Methods* 2021;**18**:1128–31.