# RAPPPID: Towards Generalisable Protein Interaction Prediction with AWD-LSTM Twin Networks

Joseph Szymborski[1,2] and Amin Emad[1,2,3,*]

[1] Department of Electrical and Computer Engineering, McGill University, Montréal, QC, Canada

[2] Mila, Quebec AI Institute, Montréal, QC, Canada

[3] The Rosalind and Morris Goodman Cancer Institute, Montréal, QC, Canada

* Corresponding Author:

Amin Emad

755 McConnell Engineering Building

3480 University Street

Montréal, QC, Canada, H3A 0E9

Email: amin.emad@mcgill.ca

## ABSTRACT

**Motivation:** Computational methods for the prediction of protein-protein interactions, while important tools for researchers, are plagued by challenges in generalising to unseen proteins. Datasets used for modelling protein-protein predictions are particularly predisposed to information leakage and sampling biases.

**Results:** In this study, we introduce RAPPPID, a method for the Regularised Automatic Prediction of Protein-Protein Interactions using Deep Learning. RAPPPID is a twin AWD-LSTM network which employs multiple regularisation methods during training time to learn generalised weights. Testing on stringent interaction datasets composed of proteins not seen during training, RAPPPID outperforms state-of-the-art methods. Further experiments show that RAPPPID's performance holds regardless of the particular proteins in the testing set and its performance is higher for biologically supported edges. This study serves to demonstrate that appropriate regularisation is an important component of overcoming the challenges of creating models for protein-protein interaction prediction that generalise to unseen proteins.

**Availability and Implementation:** Code and datasets are freely available at https://github.com/jszym/rapppid.

**Contact:** amin.emad@mcgill.ca

**Supplementary Information:** Online-only supplementary data is available at the journal's website.

## INTRODUCTION

Interactions of proteins with other proteins and their surroundings are fundamental to the internal machinery of a cell. The interaction of proteins with other proteins is of particular interest, as it is essential for a bevy of diverse cellular functions: from organising cell structure to generating metabolic energy (Huttlin *et al.*, 2017). These interactions are typically validated with a high degree of confidence by the many biological assays commonly employed today, each with their own specific advantages and challenges (Snider *et al.*, 2015). Assays for validating protein interactions range from the venerable yeast two hybrid (Y2H) (Vidal and Fields, 2014) which researchers have relied on for the past decades, to more recent Biotin-related techniques such as BioID-MS (Roux *et al.*, 2012). A characteristic of all these assays, however, is that they are costly in terms of time, labour, and materials.

Computational approaches to predict protein-protein interactions (PPIs) are therefore useful to help towards reducing the number of costly experiments researchers are required to perform. Researchers have deployed many diverse approaches to solve the task of protein sequence-based interaction prediction. Most sequence-based methods rely on the understanding that coevolution and coexpression of proteins are both tied to protein interaction and sequence similarity (Cong *et al.*, 2019; Jansen, 2003). Some methods rely on substitution matrices for sequence alignment such as BLOSUM or PAM in combination with machine learning methods to predict interactions (Henikoff and Henikoff, 1992; Ding *et al.*, 2016). Other methods utilise Support Vector Machines (SVMs) with kernels specifically designed for use with protein sequences (Ben-Hur and Noble, 2005). Other statistical methods including naïve bayes (NB) and *k*-nearest neighbors (kNN) have been used to predict protein interactions from protein sequences (Browne *et al.*, 2007). Some of

the most successful PPI prediction methods belong to the family of methods that rely on novel substring search algorithms (Li and Ilie, 2017; Dick *et al.*, 2020), operating similarly to sequence search tools like BLAST (Altschul *et al.*, 1990). Deep learning models have also been designed for predicting protein interactions (Chen *et al.*, 2019). These deep approaches commonly either learn wide networks or share weights in a twin design; the latter being shown to be both more efficient and effective (Richoux *et al.*, 2019).

Such methods, however, face many challenges due to the nature of the data on which they train. Arguably the most pervasive of which is the ability of models to generalise and predict the interactions of proteins previously unseen by the prediction method. To ensure such generalisability, careful cross-validation techniques must be used to avoid data leakage. While the necessity of appropriate cross-validation techniques is not unique to this area of research, the application of these networks (e.g., PPIs, transcriptional regulatory networks) to obtain biological insights makes it particularly important to address this challenge in the task of biological network reconstruction (Park and Marcotte, 2012; Tabe-Bordbar *et al.*, 2018).

For PPI reconstruction, developing generalizable models prove particularly difficult. The nature of PPI networks makes it easy to create datasets with testing/training splits which leak information, resulting in inflated performance metrics that cannot properly assess the generalisability of these methods. In particular, simply splitting interaction datasets into training and testing sets using random selection of edges results in the construction of testing datasets that are almost entirely comprised of interactions between proteins found in the training set. Indeed, Park and Marcotte in 2012 found that all the PPI prediction models they surveyed were tested on such naïvely

79 constructed datasets (Park and Marcotte, 2012). The surveyed models, which had optimised their
80 performance on these naïve datasets that suffered from a large degree of information leakage, were
81 also found to incur precipitous falls in their prediction metrics when tested on datasets where no
82 proteins in the testing set occurred in the training dataset.
83
84 When faced with obstacles in the construction of generalisable models, strategic and targeted
85 applications of regularisation at training time can significantly improve results. This is particularly
86 relevant in the context of deep learning methods which pay for their expressiveness by learning an
87 outsized number of parameters. Among the most common regularisation techniques used in the
88 deep learning context is "dropout" (Srivastava *et al.*, 2014). Applying dropout to a layer consists
89 of randomly zeroing the activations of the previous network with some probability $p$. However,
90 choice of regularisation techniques must be chosen with care and in accordance with the
91 architecture. For example, applying dropout directly to the hidden state of Recurrent Neural
92 Networks (RNNs) (Lipton *et al.*, 2015), an architecture which lends itself naturally to sequential
93 inputs such as amino acid chains, impairs its ability to retain its memory of previous inputs
94 (Zaremba *et al.*, 2015).
95
96 Applying regularisation to RNNs requires additional considerations. Recent work by Merity *et al.*
97 has demonstrated that randomly zeroing the weights of RNNs ("dropconnect") (Wan *et al.*, 2013)
98 rather than their hidden state activation ("dropout") effectively reduces testing error (Merity *et al.*,
99 2017). Merity *et al.* describe applying dropout to the embedding layer as well as using an averaged
100 optimiser (NT-ASGD) as part of a series of regularisation techniques dubbed Averaged Weight-

Dropped Long Short-Term memory (AWD-LSTM). The regularisation techniques used by AWD-LSTM models are specifically selected for their suitability in the context of training RNNs.

To meet the generalisation challenges posed by PPI prediction tasks, we developed a method called the Regularised Automatic Prediction of Protein-Protein Interactions using Deep Learning, or RAPPPID. RAPPPID addresses the challenges in creating generalised models for PPI prediction by adopting (with modification) the AWD-LSTM, a regularised recurrent neural network training routine (Merity *et al.*, 2017). In this study, we show that RAPPPID outperforms state-of-the-art PPI prediction methods on strict validation datasets constructed in accordance with guidelines set out by Park & Marcotte (Park and Marcotte, 2012).

## METHODS

### PPI prediction using AWD-LSTM Twin Networks

RAPPPID is first trained by considering pairs of amino acid sequences of proteins along with a label indicating whether they do or do not interact. The amino acid sequences are first tokenised using the Sentencepiece algorithm (Kudo and Richardson, 2018), which allows for better recognition of common groupings of amino acid residues that make up the secondary structure and motifs of proteins. Fixed-length latent vector representations of the token sequences of both proteins are then computed by twin neural networks (Bromley *et al.*, 1993), forming the encoder of RAPPPID's pipeline. These twin networks have shared architectures and weights and are trained jointly. An overview of the pipeline is provided in Figure 1.

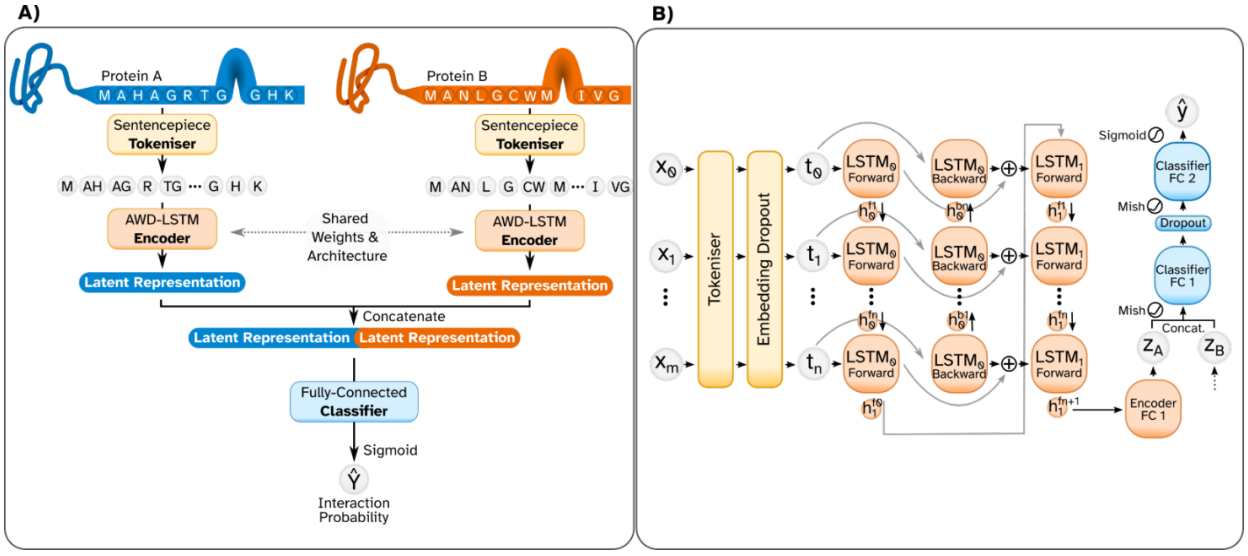



**Figure 1: Overview of the RAPPPID pipeline and architecture.** (A) The pipeline of the RAPPPID begins with two protein sequences. Each sequence is first tokenised by the Sentencepiece tokeniser (yellow). Each sequence of tokens is then inputted separately into an AWD-LSTM encoder layer which results in a latent representation for each protein. The latent representations of each protein are inputted into a fully-connected classifier layer. The classifier layer outputs the predicted probability of the two proteins interacting. (B) Taking a closer look at the architecture of the RAPPPID, individual residues $x_0$ to $x_m$ for a protein comprised of $m$ residues are tokenized into $n$ tokens $t_0, t_1, \ldots, t_n$. The embedding dropout layer randomly assigns random tokens from the total vocabulary to zero. The encoder layer is comprised of a multi-layer bidirectional LSTM whose last hidden state is fed to a fully connected layer before outputting a latent representation $z_A$. $z_A$ is then concatenated with the latent representation of a second protein ($z_b$) before being inputted into the two-layer fully-connected classifier. The output of the classifier is activated by the sigmoid function to produce a probability of interaction.

Each twin network consists of a two-layer bidirectional AWD-LSTM network (Merity *et al.*, 2017), which takes a tokenised amino acid sequence as its input and generates a fixed-length latent vector representation as its output. AWD-LSTMs are architecturally identical to the LSTMs (Hochreiter and Schmidhuber, 1997), however at inference time, several regularisation techniques are employed while training to promote learning generalised weights. Among these regularisation techniques are an averaged optimizer and dropout applied to embeddings and LSTM weights (Athiwaratkun *et al.*, 2019; Wan *et al.*, 2013). Using an AWD-LSTM encoder enables RAPPPID

to leverage the strong inductive biases of LSTMs, while ensuring that the learned weights are generalised.

The final hidden state of the AWD-LSTM is passed to a single fully-connected layer whose output, once activated by the Mish function (Misra, 2020), is the latent representation of the amino acid sequences. The latent representations of both proteins are then concatenated and provided as inputs to the classifier network which generates an interaction probability for each protein pair. The classifier network is a two-layer fully-connected network that outputs a single logit whose sigmoid activation serves as the probability of the two proteins interacting. The activated logit is then used to calculate the mean binary cross-entropy loss. Relegating the pairwise comparison of proteins to the shallower classifier network allows RAPPPID to infer protein interactions in an efficient manner. Figure 1 provides an overview of RAPPPID's pipeline.

**Sequence Segmentation and Tokenisation**

As mentioned earlier, RAPPPID utilises the Sentencepiece algorithm (Kudo and Richardson, 2018) to tokenize amino acid sequences. While words form the basis of many natural languages and may break up sentences and phrases into discrete units, no such higher-order segmentation is as immediately apparent in amino acids. Motifs and protein domains possess many analogous qualities to words in natural languages; they appear repeatedly in amino acid sequences and their combination and relative position in these sequences play important roles in the protein structure and function (Anfinsen, 1973).

Much like words, however, motifs and protein domains present an "out-of-vocabulary" problem, where unseen examples are difficult to handle. Attempts to solve this problem in natural language processing tasks has resulted in "subword" segmentation algorithms, particularly in difficult-to-segment languages such as Japanese which do not separate words by spaces (Schuster and Nakajima, 2012). Here, we employ the Sentencepiece algorithm (Kudo and Richardson, 2018) to sample tokens from "subword" vocabularies generated by the Unigram algorithm (Kudo, 2018). The unigram and sentencepiece algorithms construct vocabularies of arbitrary size by modelling the probabilities of subwords and provides a principled manner for sampling from this distribution to reconstruct sequences. The multi-residue tokens that comprise the vocabulary subdivide low-entropy areas and reduce the overall length of the sequences encoded.

**Generalising Protein Sequence Encoding with AWD-LSTM**

The task of protein-protein interaction prediction on unseen proteins is a difficult problem prone to overfitting, as demonstrated by the poor testing performance of various methods on unseen proteins (Park and Marcotte, 2012). For this reason, a training and optimisation methodology that allows efficient regularisation is desirable. AWD-LSTM was recently devised to enable efficient training of generalisable recurrent neural networks (RNNs) (Merity *et al.*, 2017). This approach deploys several regularisation techniques during training to achieve this goal. RAPPPID adopts, with modification, the training methodology of AWD-LSTM.

RAPPPID utilises Embedding Dropout, DropConnect (Wan *et al.*, 2013), and Weight Decay (Loshchilov and Hutter, 2019) on the LSTM weights, as described by AWD-LSTM, in the encoder during training. Both AWD-LSTM and RAPPPID optimise over average weights, however the

191 optimisers are quite different. AWD-LSTM makes use of the non-monotonically triggered
192 averaged stochastic gradient descent (NT-SGD) optimiser which switches between stochastic
193 gradient descent (SGD) and the averaged variant (ASGD). RAPPPID uses the recent Stochastic
194 Weight Averaging (SWA) strategy in combination with the Ranger21 optimiser (Athiwaratkun *et*
195 *al.*, 2019; Wright and Demeure, 2021).
196
197 SWA has been shown to promote generalisable models in part by overcoming the challenges of
198 finding best solutions within flat loss basins (Izmailov *et al.*, 2019). Ranger21 is an optimiser that
199 applies the "Lookahead mechanism" (Zhang *et al.*, 2019) to the AdamW optimiser (Loshchilov
200 and Hutter, 2019) and includes several optimisation techniques (Wright and Demeure, 2021).
201 These techniques, which include gradient centralisation and adaptive gradient clipping, enable us
202 to further improve our ability to learn generalised weights and smooth training trajectories (Brock
203 *et al.*, 2021; Yong *et al.*, 2020). Finally, since RAPPPID does not rely on the timestep outputs of
204 the LSTM network, the Temporal Regularisation (TAR) described by AWD-LSTM is not
205 applicable.
206
207 **Details of RAPPPID's architecture and hyperparameter tuning**
208 The dimensionality of the AWD-LSTM hidden state is made equal to the dimensionality of the
209 embeddings (*i.e.*: 64). The output of the fully-connected layer is of equal dimensionality of the
210 AWD-LSTM hidden state and activated by the Mish activation function (Misra, 2020). The output
211 of the first fully-connected layer is half the size of the embedding dimension, activated by the Mish
212 function, and regularised by a dropout layer (Srivastava *et al.*, 2014). RAPPPID trains on a

vocabulary of 250 tokens that is generated by the Sentencepiece algorithm. New vocabularies are generated for each dataset before training.

The number of LSTM layers, L2 coefficient, and various dropout rates are defined as hyper-parameters that require tuning between different datasets. The final values selected for each of these hyper-parameters are selected using a validation set, randomly selected from the training set. The range of considered hyperparameters and their selected values are provided in the Supplementary Tables S1-S3 (in Supplementary File 1).

**Datasets**

We obtained protein-protein interactions (PPIs) and protein sequences from version 11 of the Search Tool for the Retrieval of Interacting Genes/Proteins (STRING) database (Szklarczyk *et al.*, 2019), from the official STRING website. Edges were downloaded from https://stringdb-static.org/download/protein.links.detailed.v11.0.txt.gz and sequences were downloaded from https://stringdb-static.org/download/protein.sequences.v11.0.fa.gz. In this dataset, the association between any two proteins is assigned a confidence score depending on the source of the information (called a "channel"). In our analysis, only associations with a combined STRING-score above 95% (equivalent to a score above 950, obtained from combining different channels) were considered as positive edges. We also retained the channel-specific scores for further analysis.

Due to the lack of an authoritative source of pairs of proteins which are known not to interact with one another, a methodology must be adopted for generating such "negative edges". Negative edges

were randomly selected from the set of protein pairs not known to interact. While it is common for methods to generate negative examples by pairing proteins which are associated with distinct subcellular compartments from one another, evidence has shown that the naïve approach adopted here results in less bias (Ben-Hur and Noble, 2005).

**Training, Validation and Testing Set Construction**

As identified by Park and Marcotte (Park and Marcotte, 2012), methods which consider the interaction of proteins in a pairwise fashion must (and have historically failed to) take additional care to avoid information leakage when constructing training and testing datasets. Following their suggestion, here we use three different classes of testing and training sets to evaluate the performance of RAPPPID and other algorithms (Figure 2).

1) "C1" refers to the evaluation scheme in which edges (pairs of proteins) are randomly selected to form the training or testing sets. Since the selection criterion is based on edges, both proteins in the pair may be present in both the testing and training sets (due to the presence of other edges adjacent to each protein).

2) "C2" refers to the evaluation scheme in which proteins are randomly selected to form the training or testing sets. In this scheme, only one protein in a pair may be present in both testing and training sets (but never both). This evaluation scheme mimics the scenario in which a model trained on the interactome is used to predict the interaction of known proteins with a newly discovered protein (that was not used to train the model).

3) "C3" refers to the evaluation scheme in which proteins are randomly selected to form the training or testing sets. However, unlike C2, proteins which appear in the training set never appear in the testing set. This is the most strict evaluation scheme.

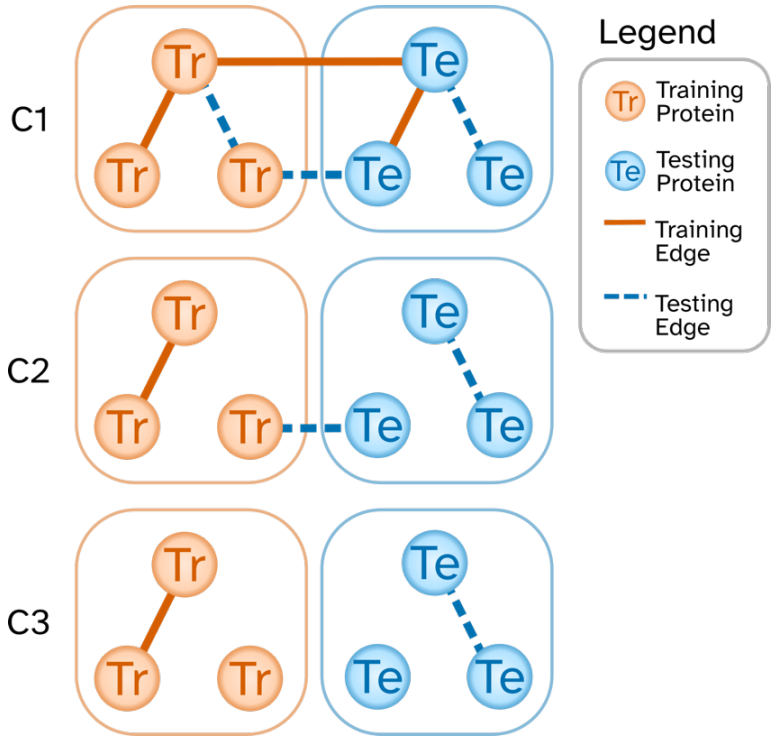



**Figure 2: Illustration of differences in edges between C1, C2, and C3 datasets.** Differences between C1, C2, and C3 datasets are most visible by first dividing the population of all proteins in the dataset into training (orange, left) and testing (blue, right). In the case of the strict C3 dataset (bottom row), edges known at training time (orange, solid) only occur between training proteins. Similarly, C3 datasets are evaluated on testing edges (blue, dotted) that only occur between testing proteins. The C2 dataset has all the edges present in the C3 dataset, but also includes testing edges between testing proteins and training proteins. Finally, the pervasive C1 datasets allows all possible training and testing edges that are not identical.

While most methods have typically reported on models validated with datasets in the C1 class, they often perform much worse on similar datasets in the more conservative C2 and C3 class. This is likely due to the information leakage between training and testing sets present in the C1 class and, to a lesser extent, in the C2 class. In our evaluations, we report the performance of different

methods using all three evaluation schemes above, but we are most interested in the results of C3 due to a lack of information leakage.

## Implementation

RAPPPID was implemented in the Python computer language using the PyTorch and PyTorch Lightning deep learning framework (Paszke *et al.*, 2019; Falcon *et al.*, 2020). Embedding Dropout and DropConnect implementation were obtained from the AWD-LSTM code base (Merity *et al.*, 2017). The source code for RAPPPID can be found by visiting https://github.com/jszym/rapppid.

## Protein Similarity Experiments

In the analysis of the sequence similarity between testing and training proteins, "Percent Identity" was measured between proteins using NCBI's PSI-BLAST tool running locally as part of version 2.12.0 of NCBI's BLAST+ software suite (Altschul *et al.*, 1997). In addition to the Percent Identity, for two proteins to be considered similar, an E-value cut-off of at most 5 and an alignment length of more than 30% of the query sequence were considered necessary. The 64-bit Linux binaries of the BLAST+ suite were obtained from the link ftp.ncbi.nlm.nih.gov/blast/executables/blast+/2.12.0/.

# Results

## Performance evaluation of RAPPPID and other algorithms

To establish the ability of RAPPPID to correctly predict protein-protein interactions within the current landscape of PPI prediction methods, we compared it against three recent methods (Figure 3). The first of these is the Scoring Protein INTeractions (SPRINT) method, which belongs to the

family of methods that predict interactions according to measures of sequence similarity (Li and Ilie, 2017). SPRINT was shown to outperform support-vector machine (SVM), random-forest (RF), and sequence similarity-based methods across C1, C2, and C3-like datasets.

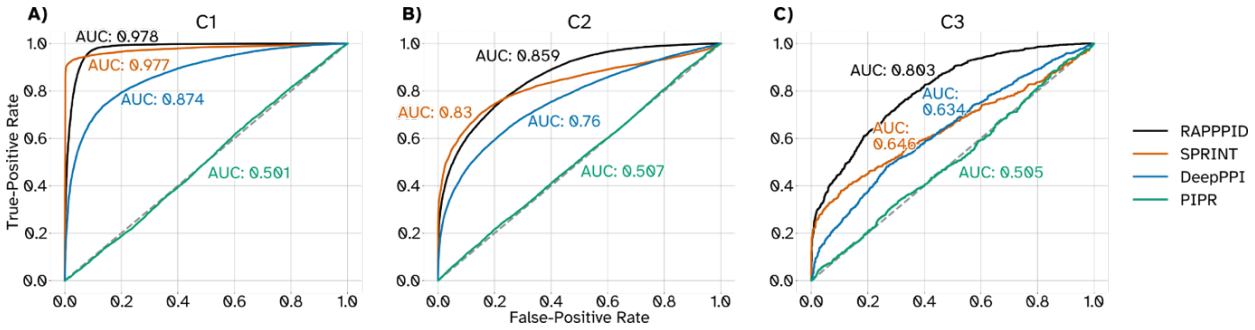



**Figure 3. Receiver-Operator curves across methods and datasets.** The receiver-operator curves (ROCs) for all four methods tested across C1 (A), C2 (B), and C3 (C) datasets.

The two other methods, PIPR and DeepPPI, are deep learning methods that similar to RAPPPID utilize twin networks (Chen *et al.*, 2019; Richoux *et al.*, 2019). PIPR uses a residual recurrent convolutional neural network (RCNN) for its encoder with the goal of more effectively summarising both local and global features. We compared RAPPPID against the best performing iteration of DeepPPI, whose encoder comprises of a convolutional neural network feature extractor followed by an LSTM network.

Across C1, C2, and C3 testing datasets, RAPPPID achieved higher area under the receiver-operator curve (AUROC) than all other methods tested (Table 1). The margin between RAPPPID and the second highest performing method (SPRINT in all cases) was highest when performed on the stricter C3 dataset, resulting in approximately a 24.3% improvement. The improvement

317 obtained by RAPPPID compared to SPRINT was lower on the C2 dataset (approximately 3.4%),
318 and finally nearly equivalent on the least strict C1 dataset.
319
320 **Table 1: Comparison of PPI prediction performance on C1, C2, and C3 datasets.** The testing
321 AUROC and AUPR of four different PPI prediction methods is reported across the three different
322 dataset types described by Park & Marcotte (Park and Marcotte, 2012).

| Dataset | Method | Testing AUROC | Testing AUPR |
|---|---|---|---|
| C1 | RAPPPID | **0.978** | 0.974 |
| | SPRINT | 0.977 | **0.983** |
| | DeepPPI | 0.874 | 0.881 |
| | PIPR | 0.501 | 0.405 |
| C2 | RAPPPID | **0.859** | **0.868** |
| | SPRINT | 0.830 | **0.868** |
| | DeepPPI | 0.760 | 0.787 |
| | PIPR | 0.507 | 0.508 |
| C3 | RAPPPID | **0.803** | **0.810** |
| | SPRINT | 0.646 | 0.716 |
| | DeepPPI | 0.574 | 0.590 |
| | PIPR | 0.505 | 0.509 |

323
324 With regards to the area under the precision-recall curve (AUPR), this trend across dataset types
325 persisted. RAPPPID's AUPR was higher than all other methods for the C3 dataset, with a margin
326 to the second highest method of 0.094 (equivalent to an approximately 14.6% improvement).
327 RAPPPID's AUPR score was matched by SPRINT in experiments conducted on the C2 dataset,
328 but outperformed DeepPPI and PIPR. SPRINT achieved the highest AUPR of all the methods in
329 the C1 dataset, outperforming RAPPPID with a margin of 0.009 (equivalent to an approximately
330 0.9% improvement).
331

While we were able to replicate results of the PIPR model on the *S. cerevisiae* dataset published as part of the original PIPR publication (Chen *et al.*, 2019), PIPR suffered from convergence issues during training on our *H. sapiens* STRING datasets. We suspect this is due to a variety of factors, with the large differences in dataset characteristics being the most likely cause. The number of proteins and interactions in the *S. cerevisiae* dataset is far smaller than our STRING datasets. Furthermore, the *S. cerevisiae* dataset selected pairs of proteins which occupy different subcellular compartments in order to construct negative samples; an approach found to lead to biased estimations of prediction accuracy (Ben-Hur and Noble, 2006).

Taken together, these results suggest that RAPPPID outperforms alternative methods in the majority of evaluations on C1, C2, and C3 schemes and is particularly effective in the stricter and more difficult C3 evaluation.

**Channel-specific performance of RAPPPID**

The STRING database, integrates and annotates protein association data from a wide range of sources. The "database", "text-mining", "experiments", and "coexpression" channels make-up the majority of the edges in our datasets (e.g., 98.4% of all the edges in the C3 dataset).

The "database" channel is comprised of several curated databases of interactions such as KEGG and Reactome (Kanehisa, 2000; Jassal *et al.*, 2019). Edges in the "text-mining" channel are the result of a statistical analysis of proteins whose names and/or identifiers co-occur in publications. The "experiments" channel is populated by interactions evidenced by high-throughput experiments curated by members of the International Molecular Exchange (IMEx) consortium

(Orchard *et al.*, 2012). This includes datasets such as IntAct, DIP, the BioGRID, and many others (Orchard *et al.*, 2014; Salwinski *et al.*, 2004; Oughtred *et al.*, 2020). Finally, the "coexpression" channel arises from proteomic and transcriptomic assays which quantify gene-by-gene correlations. STRING additionally assigns calibrated confidence scores to each of the edges which summarise the evidence supporting an edge. Scores are assigned to each edge by channel, and finally harmonised into a final "combined confidence score" which represents the evidence across all channels present in STRING."

To better characterise the results of our protein-protein prediction tests, we sought to identify the source of the testing edges RAPPPID correctly and incorrectly identified. Figure 4A and Supplementary Figure S1 show that RAPPPID can accurately predict the testing set edges that have a high confidence score in biologically supported channels of co-expression, experiments, and database. However, the accuracy for the edges that have a high confidence score in the text-mining channel is inferior to the other channels. Since edges that are only supported by text-mining (but not by the other channels) are arguably the ones most prone to error, we expected RAPPPID to have an inferior performance on such edges (since the edges themselves may not be reliable). To test whether the inferior performance of RAPPPID in the text-mining channel in Figure 4A are indeed due to such edges, for a fixed threshold k ($50 \leq k \leq 95$), we divided the testing edges with a text-mining confidence score at least equal to k into two groups: a group with "experiments" confidence score at least equal to 80% and a group with "experiments" confidence score smaller than 80% (Figure 4B). Evaluating the testing edges in C1, C2, and C3 showed that for this channel, the accuracy on the former group is higher than the latter group (sometimes as large as ~22% higher, Supplementary Figure S2). Repeating the analysis for co-expression and database channels

also confirmed this trend (Figure 4B, Supplementary Figure S2). Taken together, these results suggest that the inferior performance of RAPPPID on the text-mining channel in Figure 4A is indeed due to the edges that are supported only by text-mining and not by other biologically identified channels.

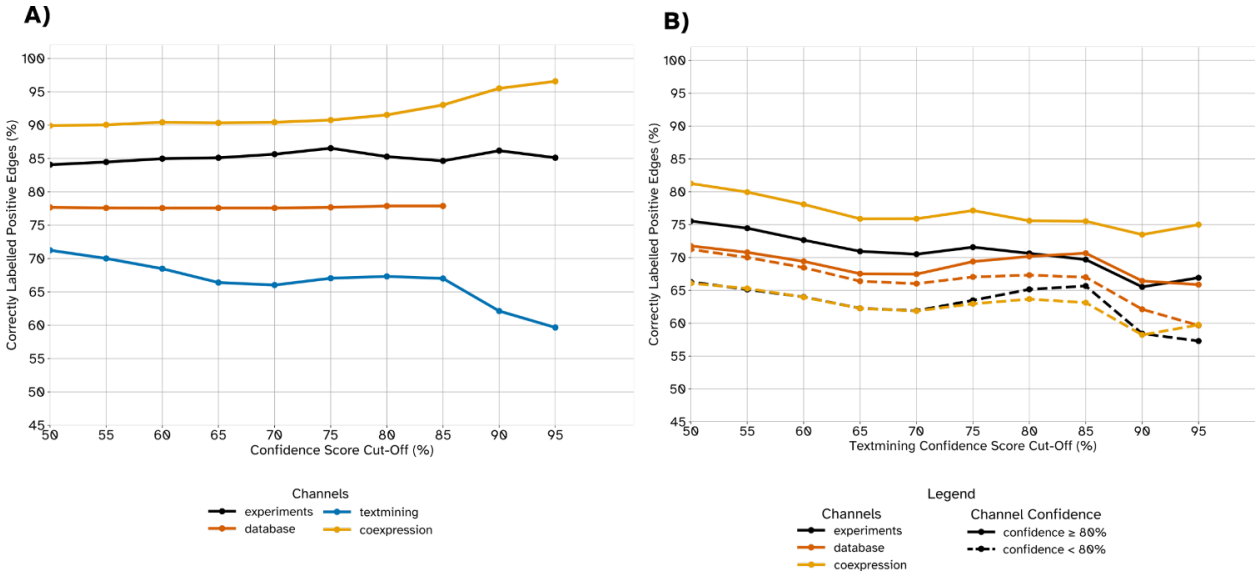
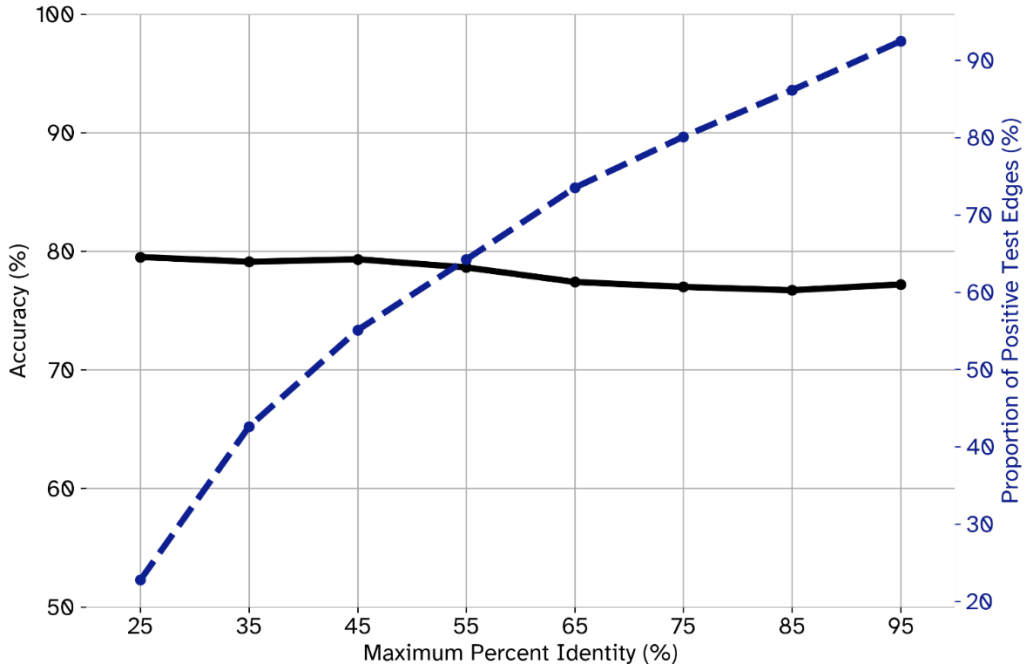



**Figure 4: Accuracy of positive edges across edge confidence stratified by STRING channels.** (A) The percentage of correctly labelled positive edges are plotted for each major STRING channel. The x-axis denotes the channel edge confidence cut-off score for each curve's respective channel. (B) Here, we see a similar chart but rather than using each channel's respective score as a confidence cut-off, edges are excluded according to their text-mining confidence (x-axis). The solid curves include edges which have a channel confidence ≥ 80% for the channel indicated by the curve's colour. Dashed curves conversely include edges whose channel confidence is < 80% for the channel indicated by the curve's colour. In both (A) and (B) data shown reflects the C2 model/dataset.

## Role of Protein Similarity on RAPPPID Performance

The procedures for C2 and C3 datasets were devised to reduce the information leakage by avoiding testing on edges which contain proteins which are known to an algorithm during its training. This safeguard against information leakage, however, does not account for proteins which are known by different identifiers but share near identical protein sequences. Further complicating the matter,

399 sequence similarity is a valid PPI prediction feature and is often used by methods as a proxy
400 measure of co-evolution and conserved functional domains (Cong *et al.*, 2019; Jansen, 2003).
401 Indeed, this strategy is leveraged by the SPRINT algorithm against which RAPPPID was
402 compared.




403
404 **Figure 5: Accuracy of positive edges as a function of similarity between testing and training**
405 **proteins in C2.** The similarity between testing and training proteins was measured using their
406 percent identity as computed by NCBI's PSI-BLAST software. The highest percent identity
407 between any training protein and a testing protein in a testing edge was considered to be that testing
408 edge's "maximum percent identity". The percentage of accurately labelled positive edges (black
409 curve, left y-axis) is reported for edges with maximum percent identities lower than the threshold
410 reported on the x-axis. The proportion of testing edges for each threshold values is reported by the
411 dashed blue curve and the right y-axis.
412

413 In spite of the challenges above, we sought to determine whether the superior performance of
414 RAPPPID (particularly in the strict datasets of C2 and C3) is due to sequence similarity between
415 testing and training proteins or not. For this purpose, we used PSI-BLAST algorithm (Altschul *et*
416 *al.*, 1997) to evaluate sequence similarities between each pair of testing/training proteins. Figure

5 shows the accuracy of RAPPPID on the C2 dataset when different degrees of restriction on sequence similarity are imposed. More specifically, a threshold *t* (x-axis in Figure 5) determined the maximum allowable Percent Identity score between a testing protein and any of the training proteins that were candidates to be similar to it (see Methods for details). Any testing protein that did not satisfy this condition for the threshold *t* was excluded from the calculation of accuracy. As one moves towards larger values of *t*, the sequence similarity constraint loosens and *t*=100% is equivalent to the complete C2 dataset. Our analysis on C2 (Figure 5) and C3 (Supplementary Figure S3) revealed that RAPPPID's accuracy is largely independent of the sequence similarities between testing and training proteins and in fact removing testing proteins that have a highly similar training protein (slightly) improves the accuracy of RAPPPID.

**Effect of stochastic components on RAPPPID's performance**

Since RAPPPID utilizes random initialisation, mini-batch sampling, and token sampling, there is some stochasticity present in its performance. To test the effect of these components and to ensure that the specific choice of training and test sets were not responsible for the superior performance of RAPPPID, we ran RAPPPID on three additional C3 datasets whose training, validation, and testing proteins were chosen at random. In each of these runs, different seeds were used to assess the effect of stochastic components of RAPPPID (mentioned above). Table 2 provides the performance metrics on the testing set as well as the validation set (which was used for selecting the hyperparameters in Supplementary Table S3). Overall, the average testing AUROC across models trained on these additional three datasets was 0.797 (±0.011). Results from these repeatability experiments illustrate that RAPPPIDs strong performance on C3 datasets is not tied to a specific set of testing or validation proteins, nor certain weight initialisation states.

**Table 2: Repeatability of results across randomly generated C3 datasets.**

| Dataset # | Validation Loss | Validation AUROC | Validation AUPR | Testing AUROC | Testing AUPR |
|---|---|---|---|---|---|
| 1 | 0.555 | 0.804 | 0.803 | 0.804 | 0.809 |
| 2 | 0.560 | 0.804 | 0.807 | 0.806 | 0.803 |
| 3 | 0.559 | 0.780 | 0.776 | 0.782 | 0.789 |

## Discussion and Conclusion

This study introduced RAPPPID, a deep learning method that addresses the challenges of creating generalisable PPI prediction models posed by inherent characteristics of PPI datasets. By adopting a modified AWD-LSTM training routine, RAPPPID was able to surpass state-of-the-art models under testing conditions that carefully controlled for information leakage and other sources of prediction accuracy inflation. Further experiments were conducted to confirm the results were independent of the specific proteins present in the training and testing splits. RAPPPIDs ability to PPIs in the STRING database was shown to increase with strong biological evidence for the interaction. This relationship between PPI evidence and RAPPPID predictive ability illustrates that RAPPPID accurately reflects our confidence in interactions, and testing performance is not disproportionately inflated by spurious, low-confidence interactions. Moreover, assessment of the sequence similarity between testing and training proteins revealed that the superior performance of RAPPPID is not due to the presence of highly similar protein pairs in testing and training, and the accuracy of RAPPPID was largely stable with a small improvement when highly similar testing proteins were excluded.

RAPPPID's ability to predict interactions warrants further study into relevant tasks that might benefit from a similar approach. The RAPPPID architecture might be modified for the tasks of

binding site and protein function prediction. These tasks are related to PPI prediction and as a result are exposed to similar challenges to which RAPPPID is well suited. However, in all these cases, it is crucial to consider strict rules for cross-validation and data splitting to ensure data leakage is avoided.

**Funding:**

This work was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2019-04460 (AE), and by McGill Initiative in Computational Medicine (MiCM) (AE). We would also like to acknowledge the Compute Canada computational resources provided to us through Mila.

**Authors' contributions:**

AE and JS conceived the study and designed the project. JS designed the algorithm, implemented the pipeline, and performed the statistical analyses of the results. All authors read and approved the final manuscript.

## References

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *Journal of Molecular Biology*, **215**, 403–410.

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

Anfinsen,C.B. (1973) Principles that Govern the Folding of Protein Chains. *Science*, **181**, 223–230.

Athiwaratkun,B. *et al.* (2019) There Are Many Consistent Explanations of Unlabeled Data: Why You Should Average. In, *ICLR*.

Ben-Hur,A. and Noble,W.S. (2006) Choosing negative examples for the prediction of protein-protein interactions. *BMC Bioinformatics*, **7**, S2–S2.

Ben-Hur,A. and Noble,W.S. (2005) Kernel methods for predicting protein–protein interactions. *Bioinformatics*, **21**, i38–i46.

Brock,A. *et al.* (2021) High-Performance Large-Scale Image Recognition Without Normalization. *arXiv:2102.06171 [cs, stat]*.

Bromley,J. *et al.* (1993) Signature verification using a "siamese" time delay neural network. *Int. J. Patt. Recogn. Artif. Intell.*, **07**, 669–688.

Browne,F. *et al.* (2007) Supervised Statistical and Machine Learning Approaches to Inferring Pairwise and Module-Based Protein Interaction Networks. In, *2007 IEEE 7th International Symposium on BioInformatics and BioEngineering*. IEEE, Boston, MA, USA, pp. 1365–1369.

Chen,M. *et al.* (2019) Multifaceted protein–protein interaction prediction based on Siamese residual RCNN. *Bioinformatics*, **35**, i305–i314.

Cong,Q. *et al.* (2019) Protein interaction networks revealed by proteome coevolution. *Science*, **365**, 185–189.

Dick,K. *et al.* (2020) PIPE4: Fast PPI Predictor for Comprehensive Inter- and Cross-Species Interactomes. *Scientific Reports*, **10**, 1390.

Ding,Y. *et al.* (2016) Predicting protein-protein interactions via multivariate mutual information of protein sequences. *BMC Bioinformatics*, **17**, 398.

Falcon,W. *et al.* (2020) PyTorchLightning/pytorch-lightning: 0.7.6 release Zenodo.

Henikoff,S. and Henikoff,J.G. (1992) Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, **89**, 10915–10919.

Hochreiter,S. and Schmidhuber,J. (1997) Long Short-Term Memory. *Neural Computation*, **9**, 1735–1780.

Huttlin,E.L. *et al.* (2017) Architecture of the human interactome defines protein communities and disease networks. *Nature*, **545**, 505–509.

Izmailov,P. *et al.* (2019) Averaging Weights Leads to Wider Optima and Better Generalization. *arXiv:1803.05407 [cs, stat]*.

Jansen,R. (2003) A Bayesian Networks Approach for Predicting Protein-Protein Interactions from Genomic Data. *Science*, **302**, 449–453.

Jassal,B. *et al.* (2019) The reactome pathway knowledgebase. *Nucleic Acids Research*, gkz1031.

Kanehisa,M. (2000) KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, **28**, 27–30.

Kudo,T. (2018) Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. *arXiv:1804.10959 [cs]*.

Kudo,T. and Richardson,J. (2018) SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Brussels, Belgium, pp. 66–71.

Li,Y. and Ilie,L. (2017) SPRINT: ultrafast protein-protein interaction prediction of the entire human interactome. *BMC Bioinformatics*, **18**, 485.

Lipton,Z.C. *et al.* (2015) A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv:1506.00019 [cs]*.

Loshchilov,I. and Hutter,F. (2019) Decoupled Weight Decay Regularization. In, *ICLR*.

Merity,S. *et al.* (2017) Regularizing and Optimizing LSTM Language Models. *arXiv:1708.02182 [cs]*.

Misra,D. (2020) Mish: A Self Regularized Non-Monotonic Activation Function. *arXiv:1908.08681 [cs, stat]*.

Orchard,S. *et al.* (2012) Protein interaction data curation: the International Molecular Exchange (IMEx) consortium. *Nature Methods*, **9**, 345–350.

Orchard,S. *et al.* (2014) The MIntAct project—IntAct as a common curation platform for 11 molecular interaction databases. *Nucl. Acids Res.*, **42**, D358–D363.

Oughtred,R. *et al.* (2020) The BioGRID database: A comprehensive biomedical resource of curated protein, genetic, and chemical interactions. *Protein Science : A Publication of the Protein Society*, **30**, 187–200.

Park,Y. and Marcotte,E.M. (2012) Flaws in evaluation schemes for pair-input computational predictions. *Nat Methods*, **9**, 1134–1136.

Paszke,A. *et al.* (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. In, Wallach,H. *et al.* (eds), *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., pp. 8024–8035.

Richoux,F. *et al.* (2019) Comparing two deep learning sequence-based models for protein-protein interaction prediction. *arXiv:1901.06268 [cs, q-bio, stat]*.

Roux,K.J. *et al.* (2012) A promiscuous biotin ligase fusion protein identifies proximal and interacting proteins in mammalian cells. *Journal of Cell Biology*, **196**, 801–810.

Salwinski,L. *et al.* (2004) The Database of Interacting Proteins: 2004 update. *Nucleic Acids Res.*, **32**, D449-451.

Schuster,M. and Nakajima,K. (2012) Japanese and Korean voice search. In, *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*., pp. 5149–5152.

Snider,J. *et al.* (2015) Fundamentals of protein interaction network mapping. *Mol Syst Biol*, **11**, 848.

Srivastava,N. *et al.* (2014) Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, **15**, 1929–1958.

Szklarczyk,D. *et al.* (2019) STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res*, **47**, D607–D613.

Tabe-Bordbar,S. *et al.* (2018) A closer look at cross-validation for assessing the accuracy of gene regulatory networks and models. *Sci Rep*, **8**, 6620.

Vidal,M. and Fields,S. (2014) The yeast two-hybrid assay: still finding connections after 25 years. *Nat Methods*, **11**, 1203–1206.

Wan,L. *et al.* (2013) Regularization of Neural Networks using DropConnect. In, Dasgupta,S. and McAllester,D. (eds), *Proceedings of the 30th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, Atlanta, Georgia, USA, pp. 1058–1066.

Wright,L. and Demeure,N. (2021) Ranger21: a synergistic deep learning optimizer. *arXiv:2106.13731 [cs]*.

Yong,H. *et al.* (2020) Gradient Centralization: A New Optimization Technique for Deep Neural Networks. In, *ECCV*.

Zaremba,W. *et al.* (2015) Recurrent Neural Network Regularization. *arXiv:1409.2329 [cs]*.

Zhang,M.R. *et al.* (2019) Lookahead Optimizer: k steps forward, 1 step back. In, *NeurIPS*.